HP 13255-90003

2645A OPERATING SYSTEM MICROCODE

Manual Part No. 13255-90003

REVISED

AUG-01-76

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

NOTE: This document is part of the 264XX DATA TERMINAL product series Technical Information Package (HP 13255).

HP 2645 FIRMWARE

INTRODUCTION

Most of the functions of the terminal are implemented in firmware rather than hardware. This reduces the complexity of the hardware and allows greater flexibility in the implementation of the various terminal functions. Additionally, functions can be modified with little, if any, impact on the hardware.

The firmware for the HP 2645A is divided up into five sections: Main Code, I/O Subsystem, Keyboard, Data Communications (Data Comm), and Alternate I/O. The 2645 uses only the first four sections. The alternate I/O section is intended for user-defined I/O devices. The I/O Subsystem section is optional with the printer subsystem and/or cartridge tape drives. Additional space is provided for extended functions required in bi-lingual terminals.

MEMORY ALLOCATION

The micro-processor used in the HP 2645A has an addressing range of 64K (0-65535). In general, the first 48K (0-48K) region is allocated to firmware, the next 4K (48-52K), for buffers; and the last 12K (52-64K), for display data. A 4K area in the firmware region (32-36K) is used for memory mapped I/O.

Firmware Allocation

Each firmware section is allocated a unique region of memory (Figure 1). The amount of memory allocation for each section is as follows:

Section	Size of Section (K)
Main Code	10
I/O Subsystem	8
Keyboard	2
Data Communications	4
Alternate I/O	2-4
Bi-lingual Code	2-4

Display Area Allocation

The upper portion of the display area is reserved for variables storage and device I/O buffers. The usage of the variables storage and I/O buffers is defined by the firmware. Part of the variables storage area is reserved for common variables (see "Common Area Allocation" section). If no buffer space is present, (48-52K), Data Comm buffers are allocated from the display area. The remaining display area is then available for display data (Figure 2). A minimum of 4K (60-64K) of display memory is included in the HP 2645A and can be expanded to 12K. The actual amount of display memory available is determined by the initialization routine in the Main Code module. The display area must be a contiguous region of memory.

Fast RAM Allocation

A 256-byte RAM area is included on each Control Memory Printed Circuit Assembly (PCA). This memory is accessed over the top plane connector between the Processor (8080A-2) and Control Memory PCA's, as are the ROM's. This eliminates the bus protocol and access contention encountered on bottom plane accesses. Thus, a higher access rate is obtained for memory references over the top plane.

The RAM on a Control Memory PCA configured for the 0-24K range is accessed with addresses in the range 110400-110777 (octal) inclusively, and a Control Memory PCA configured for 24-48K is accessed with addresses in the range 110000-110377 (octal). Only the RAM of the first Control Memory PCA is defined (Figure 3).

Two vectors are defined in the Fast RAM area: Interrupt and Display Scan. Each vector occupies three bytes. Initially, the vectors are set to the return operation code (RET) for the terminals micro-processor. A third vector is reserved for special terminal usage. This vector is not normally initialized to the return code unless special code is loaded in the terminal.

The "interrupt vector" is called each time an interrupt occurs. When called, the original A-register and program status word (PSW) are already stored on the stack and the A-register is set to the interrupt number (1-7). The interrupt vector is intially set to a return code (RET). Interrupts may be trapped by storing a jump (JMP) to a trap routine in the interrupt vector.

The display scan vector is used by bi-lingual terminals to maintain the current mode (bi-lingual/normal). A call to the locate cursor routine (RCADRA) is inserted in this vector to cause the current display enhancement variable (LSTDCD) to be updated to correspond to the current cursor location. This vector is called periodically by the monitor routine in the Main Code module.

```
+-----------------+ 0 K
|                 |
|      MAIN       |
|                 |
|      CODE       |
|                 |
+-----------------+ 10 K
|                 |
|      I/O        |
|      CODE       |
|                 |
+-----------------+ 18 K
|  KEYBOARD CODE  |
+-----------------+ 20 K
|    DATA COMM    |
|      CODE       |
+-----------------+ 24 K
|  ALTERNATE I/O  |
+-----------------+ 26 K
|    RESERVED     |
+-----------------+ 28 K
| BI-LINGUAL CODE |
+-----------------+ 30 K
                                   +----------------------+ 36 K
|     UNUSED      |
+-----------------+ 32 K        /  | SECOND FAST RAM AREA |
                               /   +----------------------+    +256
| I/O ADDRESSING  |
|     SPACE       |           /    | FIRST FAST RAM AREA  |
+-----------------+-36 K--         +----------------------+    +512
|                 |                |                      |
                                   |                      |
+-----------------+-38 K--         |                      |
|  ALTERNATE I/O  |         \      |       RESERVED       |
+-----------------+ 40 K     \     |                      |
| BI-LINGUAL CODE |           \    |                      |
+-----------------+ 42 K           +----------------------+ 38 K
|                 |
|    RESERVED     |
|                 |
+-----------------+ 48 K
|   BUFFER        |
|       SPACE     |
+-----------------+ 52 K
|                 |
|   DISPLAY       |
|                 |
|        AREA     |
|                 |
|                 |
+-----------------+ 64 K
```

Figure 1. Memory Allocation Map
================================

```
+-------------------------+  177777 (Octal)
|    COMMON VARIABLES     |     (48 bytes)
+-------------------------+  177720
|        MAIN CODE        |     (176 bytes)
|        VARIABLES        |
+-------------------------+  177440
|   KEYBOARD VARIABLES    |     (32 bytes)
+-------------------------+  177400
|        DATA COMM        |     (128 bytes)
|        VARIABLES        |
+-------------------------+  177200
|      I/O VARIABLES      |     (24 bytes)
+-------------------------+  177150
| ALTERNATE I/O VARIABLES |     (24 bytes)
+-------------------------+  177120
|     MESSAGE BUFFER      |     (80 bytes)
+-------------------------+  177000
|                         |
|                         |
|        DEVICE           |
|                         |
|          I/O            |     (512 bytes)
|                         |
|        BUFFERS          |
|                         |
|                         |
+-------------------------+  176000
|                         |
|        DISPLAY          |
|                         |
|                         |
|         AREA            |
|                         |
\\\\\\\\\\\\\\\\\\\\\\\\\\\
                                (up to 12K bytes)
\\\\\\\\\\\\\\\\\\\\\\\\\\\
|                         |
|                         |
|                         |
|                         |
|                         |
|                         |
+-------------------------+  DSPBGN
```

Figure 2. Display Area Allocation Map

```
+---------------------+ 111000
|                     |
|      KEYBOARD       |
|                     |
|      VARIABLES      |
|                     |
|      (64 bytes)     |
|                     |
|                     |
+---------------------+ 110700
|                     |
|                     |
|      DATA COMM      |
|                     |
|      VARIABLES      |
|                     |
|      (64 bytes)     |
|                     |
+---------------------+ 110600
|    ALTERNATE I/O    |
|      VARIABLES      |
| (23 bytes)  |<<<1>>>| --------  1.  Display Scan (110550-110552)
|<<<2>>>|    UNUSED   |           2.  Interrupt    (110545-110547)
+---------------------+ 110540
|                     |
|                     |
|                     |
|        STACK        |
|                     |
|                     |
|                     |
|      (96 bytes)     |
|                     |
|                     |
|                     |
|                     |
+---------------------+ 110400
```

Figure 3. Fast RAM Allocation Map
==================================

CODE MODULE DESCRIPTIONS

The following sections describe the code interface for each of the code modules in the HP 2645A: Main Code, I/O, Keyboard, Data Comm, and Alternate I/O. An overall functional description is given for each code module, followed by a specification and a general functional description for each entry point.

GENERAL SPECIFICATIONS

Constants

The firmware is divided into 2K partitions to correspond to the size of the ROM chips used in the HP 2645A. The first two and last three bytes of each partition contain pre-defined values.

The first two bytes are used to identify the ROM version and to verify that the ROM is loaded in the correct location. The first byte contains a value from "P" to "_" (01010000 to 01011111). The upper four bits are always 0101 and the lower four bits represent the version number (0-15). The second byte is set to the most significant eight bits of the 16-bit address corresponding to that location. By comparing the second byte to the actual address used to reference the location, the diagnostic can determine if the ROM is properly loaded.

The last three bytes contain the 16-bit CRC-16 remainder and the checksum for the bit pattern of the ROM. The checksum is contained in the last byte. The checksum is used by the self-test routine to verify the bit pattern of the ROM. The CRC remainder is used by the terminal diagnostic as another check on the bit pattern of the ROM.

Entry Vectors

Entry into each of the modules is made through vectors stored in the lower address portion of the module's address space. In general, these vectors consist of "jumps" (JMP) into routines within the module.

Subroutine Specifications

The subroutine descriptions will specify the parameters to be included in the registers on entry and the register results on exit. Any registers not specified in the exit list retain their entry values. The processor flags (S, Z, P, and C) are generally altered by the subroutines. Certain routines use the processor flags to signify different return conditions. These conditions are specified in the subroutine descriptions.

The names given in the subroutine description headers are the actual names used within the code module. The names specified within parentheses are the names used by external code blocks to reference the subroutine.

Indentation in the subroutine headers are used to indicate values related to a specific entry or exit condition (see description for "RCADRA" in Main Code description).

The subroutine headers contain a list of the parameters to be included when the subroutine is called and a list of results when the subroutine returns to the calling routine. Various symbols are used to designate the meaning of the items in the list.

The equal sign "=" signifies that the memory location or register contains the associated value (e.g., CHAR = INPUT CHARACTER).

The symbol "=>" is used to designate an implication. That is, when the value to the left of the "=>" is specified, the condition to the right of the symbol is implied (e.g., NC => CHARACTER FOUND means that if the processor carry flag is false, the character was found).

The symbol "->" designates that a register or memory location contains the address of a specified object (e.g., GETADR -> NEXT DISPLAY BYTE means that the memory location GETADR contains the address of the "NEXT DISPLAY BYTE").

The symbol "( )" signifies the contents of the address given by the value within the parenthesis (e.g., (D,E) = NEW VALUE means that the contents of the location, whose address is contained in the D and E register pair, contains the "NEW VALUE").

The symbols "[]" specify a bit within a byte. For example, ERRFLG[DCMERR] refers to the bit identified by the label DCMERR in the byte ERRFLG. The symbols "#" and "<>" are used to indicate the "not equal" condition (e.g., A#0 means A not equal to zero). A number with the letter "B" appended specifies an octal value (e.g., 177B).

Messages

Messages returned from the subroutines are returned as pointers to message sections. The message sections consist of an ASCII message stored in ascending address order (i.e., first character in lowest address and last character in highest address). A message can be broken up into as many as eight (8) different sections.

The beginning of each section is stored in one of eight message pointers (MSGPT1-MSGPT8) in the Common Variables area. On returns with messages, all message pointers, except for MSGPT1, are to be set by the subroutine. Some routines return the address to be stored in MSGPT1 in a specified register pair (generally H and L).

The message sections will be displayed in order starting with the section pointed to by MSGPT1. The last message section must terminate with an End of Page flag (EOP - 316 octal). All other sections must be terminated by a zero byte (a byte with all bits set to zero).

HP 2645 MAIN CODE INTERFACE

INTRODUCTION

The Main Code module contains the principal driving code for the HP 2645A. This module includes code to display characters on the screen, process escape sequences, and to perform the various display functions (e.g., Home Up, Clear Display). All other code modules operate under the direction of this module. A number of entry vectors are provided for access to routines contained in the Main Code module.

The Main Code occupies the 10K region from 0 to 10K (0 to 23777 octal) in the ROM space.

INTERFACE SPECIFICATIONS

Entry Vectors

The entry vectors to the Main Code routines begin at location 100 (octal). The vectors consist of "jumps" (JMP) to the corresponding subroutine starting address:

Location	Name	Function
100	DSPMSG	Display Message
103	RSTDSP	Restore Normal Display
106	DCNUM	Accumulate Digit for Parameterized Escape Sequence
111	DCPLUS	Add Plus Sign to Parameter
114	DCMNUS	Add Minus Sign to Parameter
117	ESCEND	Terminate Escape Sequence
122	CHKLIM	Check Parameter Limits
125	CLBLXF	Clear Pending Multi-Character Transfer Flag
130	SBLXF0	Set Pending Flag for Escape Sequence Initiated Multi-Character Transfer
133	SBLXFA	Set Pending Flag for Non-Block Mode Keyboard Initiated Multi-Character Transfer
136	STRTBL	Initialize for Display Transmission
141	CURPH	Home Cursor (exclude Transmit Only fields)
144	CURPHD	Home Down Cursor
147	FRECNT	Check Number of Free Display Blocks
152	PTBLK	Add Display Block to Free List
155	CLEARL	Clear Line
160	CLEARS	Clear Display
163	FNDTB2	Set Bit in Byte
166	SDTERM	Send Block Terminator and End Transfer
171	SDTRM1	Send Block Terminator Only
174	XPUTDC	Transmit Character

177	TRMTST	Perform Terminal Self-Test
202	CHINT0	Perform Character Function
205	INITD0	Initialize for Display Tear-Apart
210	GETDSP	Get Next Display Character for Output
213	LNFEED	Perform Line Feed
216	EXPAND	Expand Display Control Byte
221	NXTCHR	Get Next Display Character in Display List
224	GETDCM	Process Data Comm Input
227	MLKSC0	Locate First Unlocked Row
232	MLKOF0	Turn Off Memory Lock
235	HANGU0	Hang Terminal on Fatal Error
240	BUFMSG	Pointer to Buffer Overflow Message
242	DCTEST	Perform Data Comm Self-Test
245	IORMGO	Go To Code in Optional ROM
250	BN2DEC	Convert 16-bit Binary to Decimal
253	BN2DE0	Convert 8-bit Byte Binary to Decimal
256	RCADRA	Locate Current Cursor Location
261	GTMODE	Check for Page Mode

Local Variables

Local variables for the Main Code routines are stored in the RAM region 177440-177717 (octal). The definition of this area is at the discretion of the Main Code subroutines.

A message buffer of 80 bytes is located in the region 177000-177120 (octal). This area is used to display error and information messages. Messages stored in this area may extend into the device I/O buffers (176000-176777 (octal)).

Fast Access RAM

No Fast Access RAM other than the stack is used by the Main Code routines.

FUNCTIONAL DESCRIPTION

General

The Main Code module can be divided into a number of major sections: the Monitor and Function Interpretation, Cursor Control, Display Memory Management, Editing Features, Format Mode Operations, and Block Mode Operation.

Monitor and Function Interpretation

The monitor receives input from the keyboard and data comm and initiates the required operation to interpret the input. The I/O code module handles input from the CTU or alternate I/U device and calls the character interpreter routine (CHINT) to process the input.

Normally, the terminal is executing a wait loop (Flowchart 1) waiting for input and monitoring the status of the cartridge tapes. When input is received, the appropriate functions are invoked to process the input. After the functions are performed, the terminal resumes the wait loop.

The terminal is essentially a table driven state machine. The actions taken are primarily derived from the characters entered into the terminal from the keyboard, data comm, or cartridge tapes; but certain keys (e.g., READ and RECORD) initiate non-table actions. These keys initiate internal terminal functions.

The address of the currently active state table is maintained in location RNGTA. Each table entry consists of four action bytes. The first two bytes represent the lower and upper boundary of characters applicable for the entry. The last two bytes are the address of either the action routine or an index table to be used for the associated input characters. The least significant byte (LSB) of the address is stored in the first byte. If the high order bit of the address is a one, the value (with the high order bit masked out) represents the starting location of the action routine to process the input. Otherwise, the value represents the base address of an index table of action routine addresses using the lower limit as the index base. Each entry in the index table consists of two bytes.

Example-

```
B15     EQU  100000B    BIT 15
*
RTABLE  EQU  *-3        NORMAL RANGE TABLE
        DEF  40B,177B   ALPHANUMERICS
        DFAD DSPCHR+B15   EXECUTE DISPLAY ROUTINE
        DEF  7B,17B     BELL,BS,HT,LF,VT,FF,CR,SO,SI
        DFAD RTB010       USE FUNCTION INDEX TABLE
        DEF  33B,33B    ESCAPE CONTROL CODE
        DFAD ESCAPE+B15   EXECUTE ESCAPE ROUTINE
        DEF  0B,177B    ALL OTHER CODES
        DFAD CHKCTL+B15   CHECK FOR BLOCK TRANSFER TRIGGER CHARACTER
*
*  INDEX TABLE FOR <BELL> THROUGH <SHIFT IN>
*
RTB010  EQU  *
        DFAD ZBELL      BELL - SOUND KEYBOARD BELL
        DFAD BCKSPC     BS - BACKSPACE CURSOR
        DFAD HTAB       HT - FORWARD TAB CURSOR
        DFAD LNFEED     LF - LINE FEED
        DFAD NOFNCT     VT - NO FUNCTION
        DFAD NOFNCT     FF - NO FUNCTION
        DFAD CRRET      CR - RETURN CURSOR
        DFAD SHFTOT     SO - SHIFT OUT
        DFAD SHFTIN     SI - SHIFT IN
```

B15 is an equate for bit 15. RTABLE is the action table. The character processing routine requires that the action table address be three less than the actual start address of the table. RTB010 is an index table to the action routine for the characters BELL through SHIFT IN.

Cursor Control

A number of routines are contained in the Main Code to perform the various cursor control operations. This includes the cursor up, down, left, right, home up, home down, and return. Another routine is included to advance the cursor after a character has been entered on the screen. None of the above routines cause new display lines to be generated. The other cursor control routines, tab, back tab, cursor addressing, and line feed, cause new display line(s) to be generated as needed.

The current cursor position is maintained in locations CURCOL and CURROW for the screen column and row respectively. Location TLINO contains the absolute row number of the top row on the screen. Thus, the absolute row number of the current row is given by CURROW+TLINO.

Location CURADR contains the RAM address of the last character processed by the Main Code routines. LSTROW and LSTCOL contain the screen row and column location of the character corresponding to CURADR. The address of the character at the current row and column may be found by calling the routine RCADDR.

Display Memory Management

The display memory is organized into a linked list of 16-byte blocks of RAM. The Display Memory Access (DMA) hardware contains enough intelligence to follow the linked list and generate the display. Individual rows are linked with both the next and previous rows, while blocks within a row are linked only in the forward direction into the next display block for the line. The characters in display memory are stored in reverse order (i.e., characters are stored from higher address to lower address) since the DMA extracts characters from memory in order of decreasing addresses. All display blocks not currently allocated for display use are maintained on a free storage linked list.

The display memory management routines handle the allocation and de-allocation of display blocks. The GTBLK routine gets a display block from the free list. If the free list is empty, the PTBLK routine is called to remove a line from the display list and add the blocks to the free list. The GTNWLN routine generates another display line and appends it to the end of the display list. A combination of these routines and numerous in-line operations are used to create, extend, and de-allocate lines.

Short lines do not require display blocks to be allocated to fill the eighty character line as a special control byte (EOL - 314 octal) is recognized by the DMA as the end of line code. This allows for a more efficient use of the display memory because memory is not wasted in padding out short lines.

A display position may actually consist of one or more non-displaying flag characters or enhancement-controls along with the displayable character. Enhancement-controls include codes to enable/disable the various video enhancements (i.e., inverse, blinking, half bright, and underline) and to switch character sets. Flags are used as firmware control codes. Typical uses are to mark the beginning of protected and unprotected fields and to mark field checking attributes. The location LSTDCD contains the current display enhancement code and location LSTFMT contains the currently active format code.

Editing Features

Four editing functions are implemented: character insert and delete, and line insert and delete. Additionally, character insert and delete may be performed with or without wrap-around.

Line insert and delete are performed by adding or deleting a line in the display list. Insert line involves creating a new line from the free list and altering the next and previous row pointers at the appropriate location in the display list to include the new line. A line delete is performed by adjusting the next and previous line pointers in the display list to skip over the line to be deleted. The deleted line is added to the free list.

Character insert and delete are performed by shifting the characters in a line through the line's linked list. If a display block becomes empty as a result of a character delete, the block is added to the free list. Insert wrap is performed by saving the characters that get shifted out of the right margin and inserting them at the left margin of the next line. This involves saving the current state of the cursor and establishing a temporary state in which the cursor is located at the left margin of the next line and executing a character insert without wrap-around. When a character is shifted out, all non-displaying codes associated with the character are included. Delete with wrap-around operates in the converse manner.

Format Mode Operation

Format Mode involves the arrangement of the display into protected, unprotected, and transmit-only fields. The transmit-only and unprotected fields may also be defined with alphabetic or numeric character type checking. The user can alter data in the unprotected and transmit-only fields. Format Mode causes many of the terminal functions to be altered:

- The home up function moves the cursor to the first unprotected field in the display list.

- The tab and back tab functions move the cursor to the first character of the next and previous unprotected field, respectively.

- The clear line and clear display functions erase only the unprotected fields. Specifically, the clear line function will clear only the current field. Remaining fields in a line are not affected by the clear line function. The clear display function clears all unprotected fields from the current cursor location through the last field in the display.

- The character insert and delete functions operate within the current unprotected or transmit-only field rather than the current row. Also, the wrap-around option is disabled.

- The line insert and delete functions are disabled.

- In Block Mode, only data in the unprotected and transmit-only fields are transmitted to the computer.

- Next and Previous Page position the cursor in the first field in the appropriate page.

- When the last character of the last field on a page is entered, a next page operation will be performed. If the field is the last field in the form, the cursor will remain positioned after the last character in the field after the field is filled. Entering characters when the cursor is past the last field will cause an automatic home up and the entered character will appear in the first character of the first unprotected field.

Block Mode Operation

While operating in Block Mode, no data is transmitted to the host computer until requested by the computer or operator. Data entered from the keyboard simply appears on the screen and is not sent to the computer as in character mode where characters are sent as the keys are hit. This allows the user to compose text on the display, edit it, and then send the data when the user is satisfied with the display.

The user presses the ENTER key to send the display to the computer. This involves the DPSEND routine to set the display data pending flag (MFLGS [SENTER]) and to optionally set the non-displaying terminator via STTERM and/or set the starting position for the display tear apart routine (GETDSP). On invoking the display transmission, INITDU sets up the initial parameters for the tear apart function and successive calls to the GETDSP routine are made to extract the display data.

SUBROUTINE SPECIFICATIONS

Each subroutine will use the registers as specified below. Any registers not specified in the exit list are returned with their contents undisturbed. The settings of the processor flags (S, Z, P, and C) are generally not retained. Certain routines use the processor flags to represent exit conditions. These conditions are listed in the subroutine headers. The labels specified for the subroutines are the actual labels used in the source code.

Display Message

```
    DSPMSG - DISPLAY MESSAGE

      ENTRY:  MSGPT1-MSGPT8 -> MESSAGE SECTIONS
              NC => ADD MESSAGE TO NORMAL DISPLAY
              C => REPLACE DISPLAY WITH MESSAGE

      EXIT :  ALL REGISTERS DESTROYED
```

This routine is called to display information or error messages. The message sections are extracted via the 16-bit address pointers stored in MSGPT1-MSGPT8 in the common area. These messages are stored in ascending order. These pointers point to the left-most character in the message section. The message sections are terminated by either a zero byte or an End of Page code (EOP - 316 octal). A zero byte termination implies that the message continues into the next message section while an EOP, signifies the end of the message. The pointer to the first message section is stored in MSGPT1 and so forth.

When a message is added to the normal display, the message begins at the current cursor location. When a message replaces the normal display, the message starts at the upper left corner of the screen. A call to the restore display routine (RSTDSP) will restore the screen to the normal display.

Messages added to the normal display are usually informational (e.g., BASIC DATA COMM SELF-TEST OK). Messages replacing the display are usually error messages (e.g., ROM ERROR).

Restore Normal Display

```
     RSTDSP - RESTORE NORMAL DISPLAY

       ENTRY:  DON'T CARE

       EXIT :  PROCESSOR FLAGS UNCHANGED
               A,H,L DESTROYED
```

This routine is called to restore the normal display after a call to DSPMSG has been made to replace the display with a message. This routine may be called even if no message is currently on display. The display will be restored to either the operating or soft key display, whichever is currently active.

Accumulate Digits for Parameterized Escape Sequences

```
     DCNUM - ACCUMULATE PARAMETER FOR ESCAPE SEQUENCE

       ENTRY:  CHAR = INPUT CHARACTER
               RADIX = RADIX OF NUMBER ACCUMULATED
               IOCSGN = SIGN OF NUMBER
               IODATA = ACCUMULATED VALUE

       EXIT :  Z TRUE
               IOCSGN = 200B IF NO SIGN FOR VALUE
                           OTHERWISE, UNCHANGED
               IODATA = (ENTRY VALUE * RADIX) + INPUT VALUE
               A,D,E,H,L DESTROYED
```

This routine is called when a digit input character has been received for a parameterized escape sequence. A 16-bit value is maintained in location IODATA. If the digit is received with no preceding sign (IOCSGN = 0), then IOCSGN is set to 200B to indicate an unsigned value.

Add Plus Sign to Escape Sequence Parameter

```
     DCPLUS - PLUS SIGN RECEIVED FOR PARAMETER

       ENTRY:  DON'T CARE

       EXIT :  A,B,H,L DESTROYED
               IOCSGN = 1

     IF A SIGN VALUE IS ALREADY SET (IOCSGN # 0), THEN THE  ESCAPE  SEQUENCE  IS
     ABORTED BY EXITING VIA "ESCEND".
```

This subroutine is called when a plus sign is received for a parameter value in a parameterized escape sequence.

Add Minus Sign to Escape Sequence Parameter

```
DCMNUS - MINUS SIGN RECEIVED FOR PARAMETER

  ENTRY:  DON'T CARE

  EXIT :  A,B,H,L DESTROYED
          IOCSGN = -1

IF A SIGN VALUE IS ALREADY SET (IOCSGN # 0), THEN THE ESCAPE SEQUENCE IS
ABORTED BY EXITING VIA "ESCEND".
```

This subroutine is called when a minus sign is received for a parameter value in a parameterized escape sequence.

Terminate Escape Sequence

```
ESCEND - END ESCAPE SEQUENCE PROCESSING

  ENTRY:  DON'T CARE

  EXIT :  RNGTA = RTABLE IF NORMAL MODE
                = DFSTB0 IF SOFT KEY MODE
          ESCFLG = 0
          MFLGS2(ESCINP) = 0
          A,H,L DESTROYED
```

This escape sequence at the end of escape sequence processing to restore normal character processing.

Check Parameter Limits

```
CHKLIM - CHECK PARAMETER BOUNDARY CONDITIONS

  ENTRY:  B = CURRENT VALUE
          C = MAXIMUM ALLOWABLE VALUE
          D,E -> PARAMETER TO BE SET
          IODATA = INPUT VALUE (2 BYTES)
          IOPSGN = -1 => NEGATIVE ADJUSTMENT
                    0 => ABSOLUTE SETTING
                   +1 => POSITIVE ADJUSTMENT

  EXIT :  (D,E) = NEW VALUE
          A,C,H,L DESTROYED
```

This subroutine is called to evaluate a parameter that alters an existing variable in the terminal (e.g., cursor address). If IOPSGN is +1, the new value is the sum of the current value (B) and the input value (IODATA). If IOPSGN is

-1, the new value is the current value minus IODATA. Otherwise, the new value is set to the input value. The new value must be in the range from zero (0) to the maximum as specified in the C-register on entry to this routine. The largest maximum value is 256. If the computed value is greater than the maximum allowed, the new value is set to the maximum allowed. Similarly, if the computed value is less than zero, the new value is zero.

Clear Pending Multi-Character Transfer Flag

```
     CLBLXF - CLEAR BLOCK TRANSFER PENDING FLAG

       ENTRY:  B = 377B-(BITS TO CLEAR FROM "MFLGS")
               C = 377B-(BITS TO CLEAR FROM "MFLGS2")

       EXIT :  H = BASEH (377B)
               A,B,L DESTROYED
```

This routine is called to clear a multi-character transfer pending flag. If no more transfers are pending, the keyboard is unlocked and the block transfer trigger flag is cleared. Otherwise, only the block transfer trigger flag is cleared and the keyboard remains locked.

The pending flags are bits stored in locations "MFLGS" and "MFLGS2":

	Bit	Function
MFLGS	0	DC2 PENDING
	1	TERMINAL STATUS PENDING
	2	ALTERNATE TERMINAL STATUS PENDING
	3	DEVICE STATUS PENDING
	4	CURSOR SENSE PENDING
	5	FUNCTION KEY PENDING
	6	DISPLAY SEND PENDING (ESC d or ENTER KEY)
	7	DEVICE DONE RESPONSE PENDING
MFLGS2	0	DEVICE RECORD PENDING
	1	BINARY DATA PENDING
	2	RELATIVE CURSOR SENSE PENDING

Set Pending Flag for Escape Sequence and Non-Block Mode

Keyboard Initiated Multi-Character Transfers

```
     SBLXFO - SET BLOCK TRANSFER FLAG FOR ESCAPE
       SEQUENCE INITIATED MULTI-CHARACTER TRANSFERS

     SBLXFA - SET BLOCK TRANSFER FLAG FOR NON-BLOCK
       MODE KEYBOARD INITIATED MULTI-CHARACTER TRANSFERS

       ENTRY:  B = FLAGS TO BE SET IN "MFLGS"
               C = FLAGS TO BE SET IN "MFLGS2"

       EXIT :  ALL REGISTERS DESTROYED
               MULTI-CHARACTER TRANSFER TRIGGER (XONFLG)
                 AND DC2 PENDING FLAG ARE SET ACCORDING
                 TO OPTION SWITCHES G AND H
```

These routines are called to set the pending flag for a multi-character transfer initiated either by an escape sequence (e.g., cursor sense) or from the keyboard while not operating in Block Mode (e.g., ENTER key). See the description for the "Clear Multi-Character Transfer Pending Flag" routine for a list of the flags.

Initialize for Display Transmission

```
     STRTBL - SET FIRST DISPLAY OUT CHARACTER FOR
       DISPLAY STORE OR TRANSMIT

       ENTRY:  DON'T CARE

       EXIT :  CURCOL, CURROW = STARTING POSITION
               ALL REGISTERS DESTROYED
```

This routine is called to set the starting location for tearing apart the display. If the Auto-Terminator option switch (J) is open, a non-displaying terminator is placed ahead of the current cursor position and a reverse scan is made for the first terminator before the current cursor position as the starting location. If no terminator is found or option switch J is closed, then the display tear apart is set to begin at the home position.

Home Cursor (exclude transmit-only fields)

```
      CURPH - HOME CURSOR (EXCLUDE TRANSMIT-ONLY FIELDS)

        ENTRY:  DON'T CARE

        EXIT :  ALL REGISTER DESTROYED
```

This routine places the cursor at the "home" location in the display memory. In non-Format Mode, the cursor is placed at the left margin of the first unlocked line in the display. Then the display is rolled down until the first unlocked line in display memory is the first unlocked line on the screen.

In Format Mode, the cursor is placed in the first unprotected field in the display after the top unlocked line is rolled down to be the first unlocked line on the screen. This may result in the cursor being placed in the locked portion of the screen if an unprotected field is included in the locked portion of the screen.

Home Down Cursor

```
      CURPHD - HOME DOWN CURSOR

        ENTRY:  DON'T CARE

        EXIT :  ALL REGISTERS DESTROYED
```

This routine places the cursor at the bottom of display memory. If the last line contains any characters, the cursor is placed one line below the last line. Otherwise, the cursor is located in the last allocated line. In any event, the cursor is positioned at the left margin. The display is rolled up as required to put the last line on the screen.

Check Number of Free Display Blocks

```
      FRECNT - CHECK NUMBER OF FREE BLOCKS

        ENTRY:  DON'T CARE

        EXIT :  Z => ENOUGH FREE BLOCKS AVAILABLE
                NZ => NOT ENOUGH FREE BLOCKS
                ALL REGISTERS DESTROYED
```

This routine is used during Edit Mode to determine if enough free blocks are available to display the next input record. A successful return (Z) is made when twenty-five (25) or more display blocks are free. If there are insufficient free blocks, a call is made to the de-allocate routine (PTBLK) in an attempt to generate more free blocks. A fail return (NZ) occurs if no blocks can be de-allocated.

Add Display Block to Free List

```
      PTBLK - RELEASE A LINE TO THE FREE LIST FROM
        THE DISPLAY LIST

      ENTRY:  DON'T CARE

        EXIT :  Z => LINE NOT RELEASED
                  NC => MEMORY LOCKED
                  C => OUTPUT FAILED FOR EDIT MODE PUT
                  ALL REGISTERS DESTROYED
                NZ => LINE RELEASED
                  D,E -> FIRST DISPLAY CHARACTER
                          IN LINE RELEASED
                  A = E
                  B,C,H,L DESTROYED
```

This routine removes a line from the display list and adds the line to the free list. The top display line is de-allocated if the cursor is in the last display line. Otherwise, the last display line is de-allocated (i.e., Memory Lock occurred or I/O failure when recording line in Edit/Data-Logging Mode).

A fail return of Z, C implies that an I/O failure occured when an attempt was made to record the de-allocated line onto an I/O device. Otherwise, the fail return is due to a Memory Lock.

Clear Line

```
      CLEARL - CLEAR LINE

        ENTRY:  DON'T CARE

        EXIT :  A = -1 => CURSOR PAST END OF PAGE (EOP), CLEAR NOT DONE
                  =  0 => CHARACTER FOUND AND CLEAR DONE
                  >  0 => CURSOR PAST EOL, CLEAR NOT DONE
                ALL OTHER REGISTERS DESTROYED
```

This routine performs the Clear Line function. In non-Format Mode, the line is cleared from the current cursor location to the end of the line. Any cleared out blocks are added to the free list.

In Format Mode, clearing terminates at the end of the unprotected or transmit-only field. Blanks are written into the field starting from the current cursor location to the end of the field. If the cursor is in a protected region of the display, the clear line function is not performed.

Clear Display

```
CLEARS - CLEAR DISPLAY FROM CURSOR POSITION

  ENTRY:  DON'T CARE

  EXIT :  ALL REGISTERS DESTROYED
```

This routine performs the clear display function. The display is cleared from the current cursor location to the end of display memory. In non-Format Mode, all lines below the current line and any cleared out blocks in the current line are added to the free blocks list.

In Format Mode, only unprotected fields are cleared. If the cursor is in a transmit-only field, the transmit-only field will also be cleared from the current cursor location to the end of the field. No other transmit only fields are cleared. If the cursor is in a protected region of the display, all unprotected fields starting with the next unprotected field, are cleared. Blanks are written into each field to clear the field. No display blocks are de-allocated unless display enhancements are removed.

Set Bit in Byte

```
FNDTB2 - SET BIT N

  ENTRY:  B =  BIT NUMBER (N) TO BE SET (1-8)

  EXIT :  A = BYTE WITH BIT N SET
          B = 0
```

This routine generates a byte with the nth bit set. The least significant bit is bit 1 and the most significant, bit 8.

Send Block End Character(s) and End Transfer

```
SDTERM - SEND BLOCK END CHARACTER(S) AND END TRANSFER

  ENTRY:  DON'T CARE

  EXIT :  A DESTROYED
```

This routine causes transmission of the block end character(s) and makes a call to the Data Comm Code module to signal the end of a block transmission.

If the terminal is operating in Block-Page Mode (BLOCK MODE key down and keyboard option switch D open), only the block terminator character (BLKTRM from the Data Comm module) is sent. Otherwise, a Return and an optional Line Feed (if the AUTO LF key is down), are sent without a block terminator character.

Send Block End Character(s) Only

SDTRM1 - SEND BLOCK END CHARACTER(S)

ENTRY: DON'T CARE

EXIT : A DESTROYED

This routine causes only the block end character(s) to be transmitted as described above for the SDTERM routine. But the Data Comm module is not called to signal the end of the transmission block.

Transmit Character

XPUTDC - TRANSMIT CHARACTER

ENTRY: A = CHARACTER TO BE TRANSMITTED

EXIT : NC => TRANSMIT SUCCESSFUL
C => CHARACTER NOT TRANSMITTED
A DESTROYED

This routine is called to transmit a character out from the Data Comm module. The character is transmitted only if the terminal is in Remote Mode. If the terminal is not in Remote Mode , the routine returns as if the character had been successfully transmitted.

Perform Terminal Self-test

TRMTST - PERFORM TERMINAL SELF-TEST

ENTRY: DON'T CARE

EXIT : ALL REGISTERS DESTROYED

This routine performs the terminal self-test operation. A return is made only if the self-test is successful. If the self-test command is initiated from the data comm (DFLGS[SDACOM] = 0), a full terminal reset will be executed if a self-test error occurs. Otherwise, the terminal will lock up (jump to routine HANGUP) with an error message displayed on the screen.

Perform Character Function

```
     CHINTO - PERFORM CHARACTER FUNCTION

       ENTRY:  C = INPUT CHARACTER

       EXIT :  Z => FAST STORE ROUTINE USED
               NZ => FULL PROCESSING USED
               ALL REGISTERS DESTROYED
```

This routine (Flowchart 2) performs the character function according to the action table pointed to by RNGTA. Fast processing adds the input character directly to the display list without scanning the action table. Normal processing searches the action table for the character function. Fast processing is used only if the previous character was a display character added to an existing display block (CRAFLG > 0) and the current character is displayable (40B <= input character < 200B).

Initialize for Display Tear-Apart

```
     INITDO - INITIALIZE FOR DISPLAY GET

       ENTRY:  DON'T CARE

       EXIT :  Z => STARTING CHARACTER FOUND
                 GETADR -> FIRST CHARACTER
               NZ => CHARACTER NOT FOUND
                 GETADR UNCHANGED
               ALL REGISTERS DESTROYED
```

If Format Mode is off, CURCOL is set to zero, otherwise, CURCOL and CURROW are set to the next unprotected character.

This routine locates and sets the tear apart pointer (GETADR) to the initial character for the "GETDSP" routine. If the terminal is in Format Mode (MDFLG1[FORMAT] = 1), GETADR is set to the address of the character at the current cursor location. Otherwise, the cursor and start address are set to the first character in the current line.

A "no character found" return (NZ) occurs if the cursor is below the last display line.

Get Next Character for Output

```
GETDSP - GET A CHARACTER FROM THE DISPLAY

  ENTRY:  GETADR -> NEXT DISPLAY BYTE
          CURCOL = COLUMN NUMBER OF NEXT DISPLAY BYTE

  EXIT :  NC => CHARACTER FOUND
            A = CHARACTER
            CURCOL, GETADR UPDATED FOR NEXT CHARACTER
          C => NO CHARACTER
            M => END OF DISPLAY
            Z => END OF FIELD
            P, NZ => END OF LINE
            A DESTROYED
          B-L DESTROYED
```

This routine gets the next display character when tearing apart the display. The character returned may be either a 7-bit ASCII character or an 8-bit display enhancement/flag character. The expand routine (EXPAND) may be called to generate the escape sequence for the enhancement/flag character.

Perform Line Feed

```
LNFEED - EXECUTE LINE FEED

  ENTRY:  DON'T CARE

  EXIT :  ALL REGISTERS DESTROYED
          SPOWL = 377B
```

This routine performs the line feed function. A new line is generated if the cursor is moved below the last display line. The space overwrite (SPOW) latch is cleared by setting location SPOWL to all ones.

Expand Display Control Byte

```
     EXPAND - EXPAND DISPLAY CONTROL/FLAG BYTE
       TO ESCAPE SEQUENCE

       ENTRY:  A,C = DISPLAY CONTROL/FLAG BYTE

       EXIT :  B2DBFL = ESCAPE SEQUENCE TO GENERATE
                             DISPLAY CONTROL/FLAG BYTE
               B2DPTR -> B2DBFL-1 (LSB OF ADDRESS ONLY)
               B2DEND -> LAST CHARACTER OF ESCAPE SEQUENCE
                            IN B2DBFL (LSB OF ADDRESS ONLY)
```

This routine accepts a display enhancement or flag character and generates the escape sequence(s) necessary to reproduce the character. The result is placed into a 9-byte buffer stored in increasing address order. Locations B2DPTR and B2DEND contain the LSB portion of the "first - 1" and last character address of the escape sequence, respectively. The MSB part of the addresses is BASEH (377B).

Get Next Character in Display List

```
     NXTCHR - GET NEXT CHARACTER IN DISPLAY LIST

       ENTRY:  D,E -> CURRENT CHARACTER

       EXIT :  Z => CHARACTER IS NOT AN EOL LINK
                 A = DISPLAY CHARACTER
                 D,E -> CHARACTER
               NZ => NEXT CHARACTER IS AN EOL LINK
                 A DESTROYED
                 D,E -> NEXT LINE LINK (MSB)
```

This routine gets the next character in the display list. Display links at the end of a display block are automatically skipped over to get the first character in the next display block.

Process Data Comm Input

```
     GETDCM - PROCESS DATA COMM INPUT IF ANY

       ENTRY:  DON'T CARE

       EXIT :  CARRY FLAG = 0 (NC)
               NZ => NO MORE DATA IN DATA COMM BUFFER
               Z => FULL PROCESSING USED => DATA COMM
                     BUFFER MAY NOT BE EMPTY
               ALL REGISTERS DESTROYED
```

This routine (Flowchart 3) gets any input from the Data Comm and processes the input. The data comm is accessed only if the terminal is in Remote mode. If fast processing is used (see description of CHINT), data continues to be fetched until either the data comm buffer is empty or full processing is required for the input.

Locate First Unlocked Row

```
     MLKSCO - LOCATE FIRST UNLOCKED ROW

       ENTRY:  DON'T CARE

       EXIT :  Z => NO UNLOCKED ROWS ON SCREEN
                 A,C,H,L DESTROYED
               NZ => FIRST UNLOCK ROW FOUND
                 H,L -> FIRST UNLOCKED ROW
                   (POINTS TO LSB OF NEXT LINE POINTER)
                 A,C DESTROYED
```

This routine locates the first unlocked line on the screen. If display lock is not enabled, the address of the top line on the screen is returned. Otherwise, the address of the first line below the display lock boundary is returned. A fail return (Z) occurs if the display lock boundary is below the last line on the screen.

Turn Off Memory Lock

```
MLKOFO - TURN OFF MEMORY LOCK

  ENTRY:  DON'T CARE

  EXIT :  MLKFLG = 0
          A,B,H,L DESTROYED
```

This routine clears the display lockout condition. If the memory lock function is set for display lock, then the memory lock function remains enabled and only the lockout condition and LED blinking are turned off. Otherwise, the memory lock function is turned off also (MDFLG1[MEMLOK] = 0).

Hang Terminal on Fatal Error

```
HANGUO - DISPLAY FAIL MESSAGE AND HANG TERMINAL

  ENTRY:  H,L -> FIRST MESSAGE SECTION
          MSGPT2-MSGPT8 = POINTERS TO REMAINING
            MESSAGE SECTIONS
```

This routine is called to display an error message and hang the terminal. The contents of the H and L registers are stored in MSGPT1 and the message is displayed via the DSPMSG routine. Then the terminal is "hung" by executing an endless loop. The terminal can be restored to normal operation only by pressing the RESET TERMINAL button or by powering the terminal off, then on.

Pointer to Buffer Overflow Message

```
BUFMSG - ADDRESS OF BUFFER OVERFLOW MESSAGE
```

This 2-byte location contains a pointer to the message "BUFFER OVERFLOW" in the Main Code module. The pointer points to the first character (B) and the message is stored from low address to high address. The message is terminated by an EOP code (316B). The address value may used as a parameter to the DSPMSG routine.

Perform Data Comm Self-test

```
     DCTEST - EXECUTE DATA COMM SELF-TEST

       ENTRY:  DON'T CARE

       EXIT :  ALL REGISTERS DESTROYED
```

This routine invokes the data comm self-test routine of the data comm code module (2DCTST). The data comm self-test routine is executed only if the terminal is in Remote mode. A return is made only if the self-test executed successfully or if the terminal is in Local mode.

Go To Code in Optional ROM

```
     IORMGO - PERFORM FUNCTION IF OPTION ROM IS PRESENT

       ENTRY:  H,L -> VECTOR TO BE ENTERED

       EXIT :  NC => FUNCTION EXECUTED
                 REGISTERS SET ACCORDING TO FUNCTION
               C => FUNCTION NOT EXECUTED
                 A DESTROYED
```

This routine is called to enter code in an optional ROM. This routine determines the presence of a ROM by inspecting the first two bytes in the ROM space to be entered. The first two bytes are accessed by setting the LSB part of the address to zero (0) and one (1). The upper four bits of the first byte must contain the pattern "0101" and the second byte must match the MSB part of the address.

Convert 16-bit Binary to Decimal

```
     BN2DEC - CONVERT DOUBLE WORD BINARY TO DECIMAL

       ENTRY:  D,E = BINARY VALUE TO BE CONVERTED
               H,L -> FIRST BYTE IN OUTPUT BUFFER

       EXIT ;  H,L -> NEXT BYTE IN OUTPUT BUFFER
               A-E DESTROYED
               LNKSAV, CNTFAD DESTROYED
```

This routine converts a 16-bit binary value to its decimal ASCII equivalent. The conversion can result in as many as five (5) ASCII characters. The output buffer is filled from the low address to a higher address. On the return from this routine, the H and L registers will contain the next available byte in the output buffer, which will be set to null (all zeroes) by the conversion routine. LNKSAV and CNTFAD are used as work areas by this routine.

Convert 8-bit Binary to Decimal

```
      BN2DE0 - CONVERT SINGLE BYTE TO ASCII DECIMAL

        ENTRY:  A = BYTE TO BE CONVERTED
                H,L -> OUTPUT BUFFER

        EXIT :  NZ
                H,L -> NEXT BYTE IN OUTPUT BUFFER
                A-E DESTROYED
                LNKSAV, CNTFAD DESTROYED
```

This routine performs the same operation as BN2DEC above, but operates only on an 8-bit value.

Locate Current Cursor Location

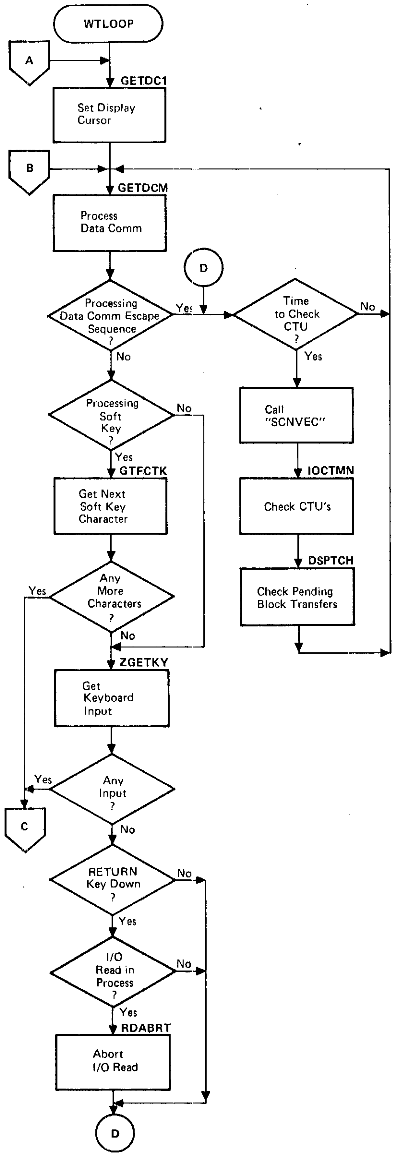
```
      RCADRA - LOCATE CURRENT CURSOR LOCATION

        ENTRY:  CURROW,CURCOL = DESIRED ROW/COLUMN POSITION
                LSTROW,LSTCOL = LAST ROW/COLUMN PROCESSED
                CURADR = ADDRESS CORRESPONDING TO LSTROW, LSTCOL
                LSTLIN -> LINE CORRESPONDING TO LSTROW

        EXIT :  Z => CHARACTER FOUND
                  D,E -> CHARACTER AT CURSOR LOCATION
                  A,B,C,L DESTROYED
                NZ => CHARACTER NOT FOUND
                  M => NEED ADDITIONAL ROWS
                    E = NUMBER OF ROWS NEEDED
                  P => ROW LOCATED
                    A = COLUMN NUMBER FOUND
                    B = ROW NUMBER FOUND
                    C = NUMBER OF CHARACTERS NEEDED
                    D,E -> LAST CHARACTER FOUND
                LSTROW,LSTCOL,LSTLIN,CURADR,LSTDCD ARE UPDATED
                  TO REFLECT THE LAST CHARACTER FOUND
                NROWS,BLKFIL = 0
```

This routine locates the address of the character corresponding to the current cursor location. A character position may consist of one or more non-displaying enhancement followed by the displayable character. When the cursor position is located, the address of the displayable character at the cursor position is returned.

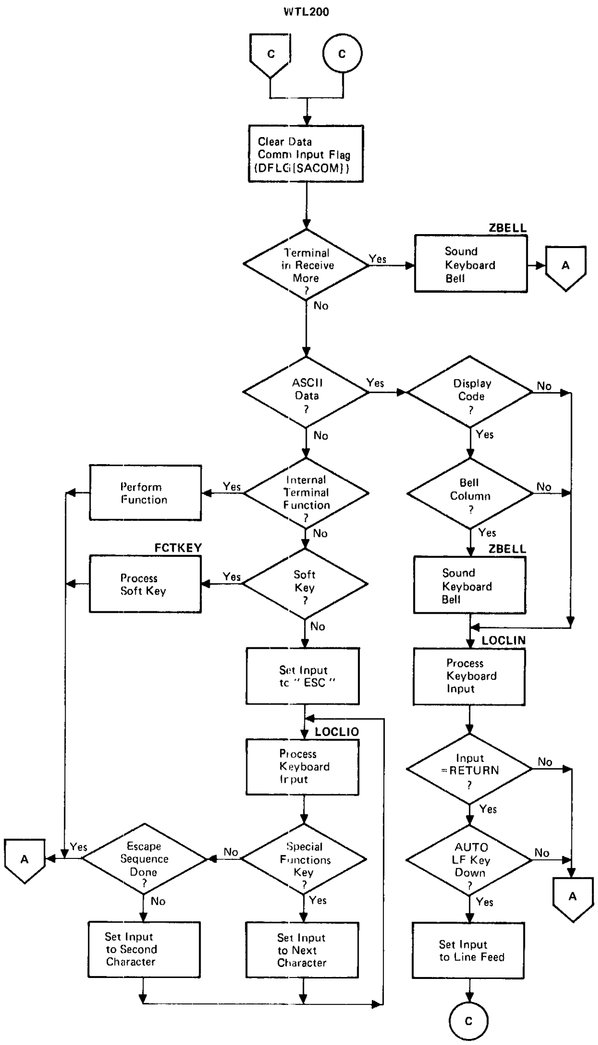
Check for Page Mode

```
     GIMODE - CHECK FOR PAGE MODE OPERATION

       ENTRY:  DON'T CARE

       EXIT :  NZ => TERMINAL IN PAGE MODE
               Z => TERMINAL NOT IN PAGE MODE
```

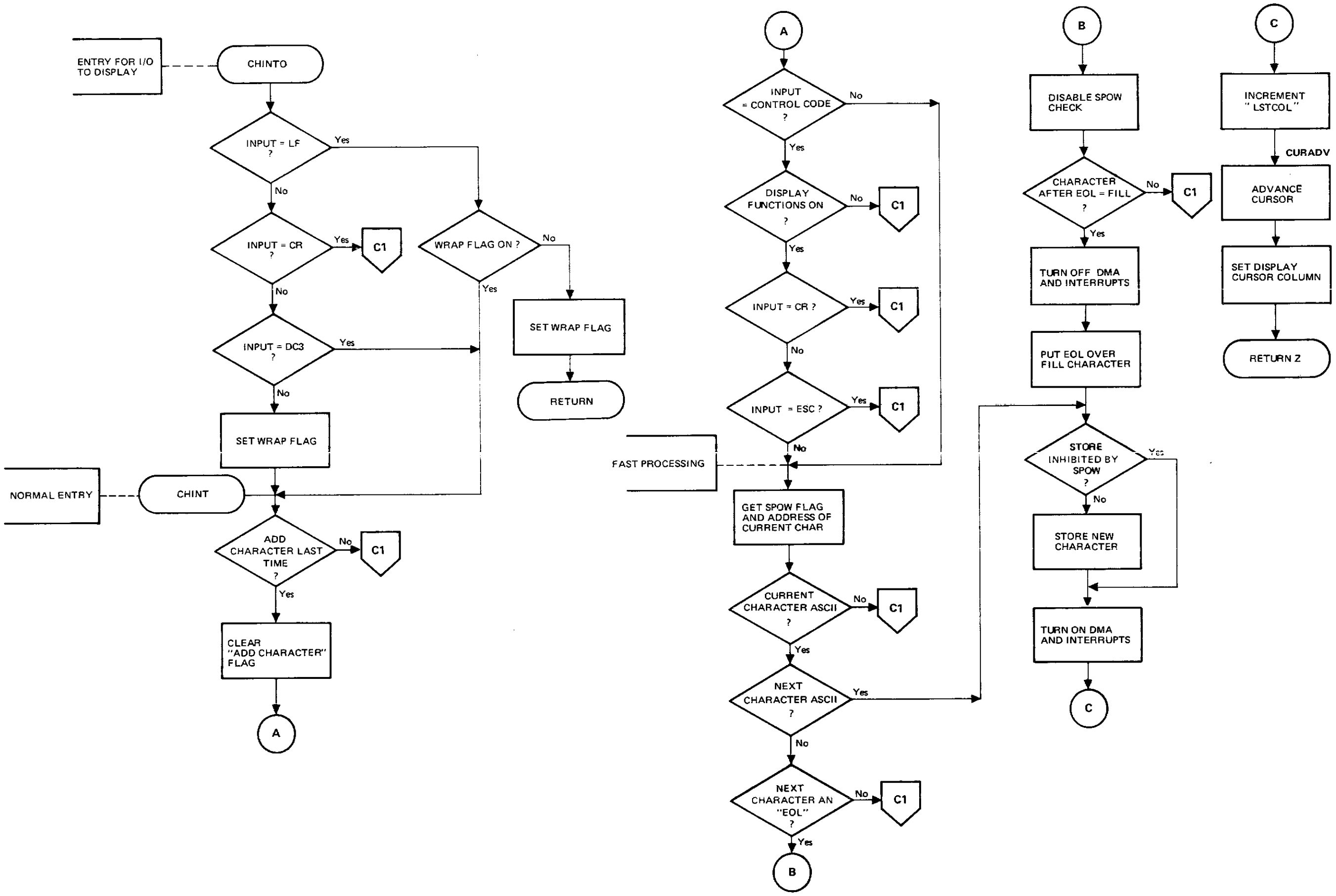
This routine determines whether or not the terminal is in Page Mode. The terminal is in Page Mode if the BLOCK MODE key is down (MDFLG2[BLKMDE] = 1) and the Page/Line option switch (D) on the keyboard interface is open (KBJMPR[PAGSIR] = 1).



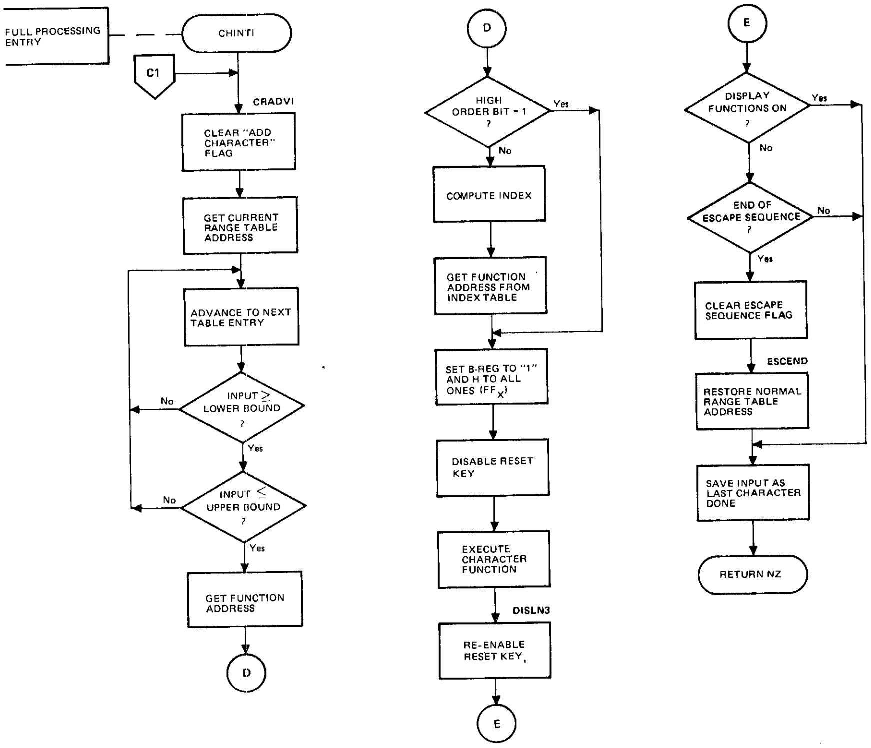
Flowchart 1A
Wait Loop Flowchart 1A
AUG-01-76   13255-90003



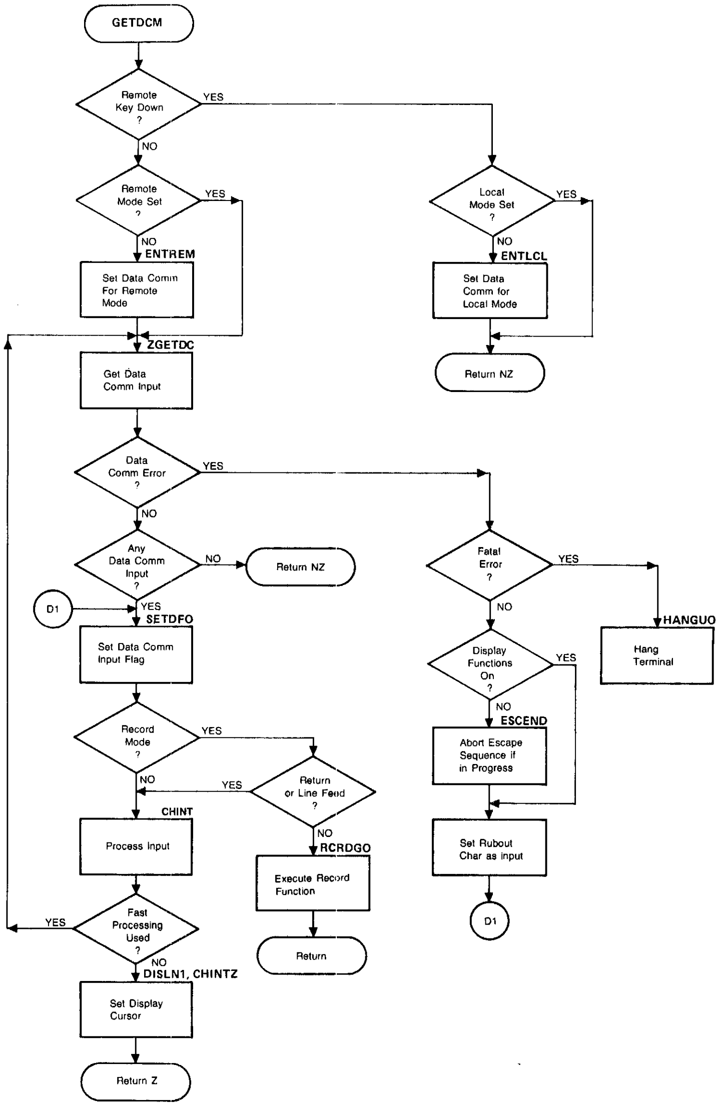
Flowchart 1B
Wait Loop Flowchart 1B
AUG-01-76 13255-90003



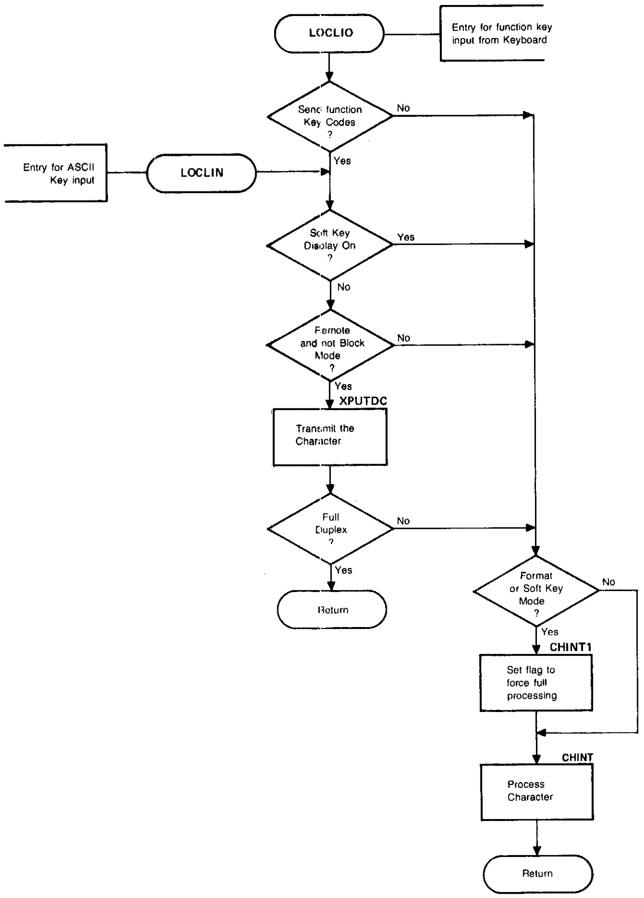
Flowchart 2A
Character Interpretation Routine Flowchart 2A
AUG-01-76 13255-90003


FULL PROCESSING ENTRY
CHINTI
C1
CRADVI
CLEAR "ADD CHARACTER" FLAG
GET CURRENT RANGE TABLE ADDRESS
ADVANCE TO NEXT TABLE ENTRY
INPUT ≥ LOWER BOUND ?
No
Yes
INPUT ≤ UPPER BOUND ?
No
Yes
GET FUNCTION ADDRESS
D
D
HIGH ORDER BIT = 1 ?
Yes
No
COMPUTE INDEX
GET FUNCTION ADDRESS FROM INDEX TABLE
SET B-REG TO "1" AND H TO ALL ONES (FF$_X$)
DISABLE RESET KEY
EXECUTE CHARACTER FUNCTION
DISLN3
RE-ENABLE RESET KEY
E
E
DISPLAY FUNCTIONS ON ?
Yes
No
END OF ESCAPE SEQUENCE ?
No
Yes
CLEAR ESCAPE SEQUENCE FLAG
ESCEND
RESTORE NORMAL RANGE TABLE ADDRESS
SAVE INPUT AS LAST CHARACTER DONE
RETURN NZ

Flowchart 2B
Character Interpretation Routine Flowchart 2B
AUG-01-76 13255-90003



Flowchart 3
Data Comm Input Processor Flowchart
AUG-01-76 13255-90003


LOCLIO
Entry for function key input from Keyboard
Send function Key Codes ?
No
Yes
Entry for ASCII Key input
LOCLIN
Soft Key Display On ?
Yes
No
Remote and not Block Mode ?
No
Yes
XPUTDC
Transmit the Character
Full Duplex ?
No
Yes
Return
Format or Soft Key Mode ?
No
Yes
CHINT1
Set flag to force full processing
CHINT
Process Character
Return

Flowchart 4
Keyboard Input Processor Flowchart
AUG-01-76 13255-90003

HP 2645 I/O CODE INTERFACE

INTRODUCTION

The I/O code module contains the logical and physical drivers to operate the I/O devices in the HP 2645A. The I/O devices include the two cartridge tape units (CTU) and the printer (RS 232, parallel or video output). Also included in this module is code to process the peripheral escape sequence (ESC & p) and the I/O function key sequences.

The I/O code occupies the 8K region from 10K to 16K (24000 to 43777 octal) in the ROM space.

INTERFACE SPECIFICATIONS

Entry Vectors

The entry vectors to the I/O routines begin at location 24002 (octal). The vectors generally consist of "jumps" (JMP) to the corresponding subroutine starting locations:

ADDRESS	NAME	FUNCTION
24002	IOCKEY	Execute GREEN Key Functions
24005	REDKEY	Handle READ Key
24010	CTLRED	Handle CONTROL READ
24013	RECKEY	Handle RECORD Key
24016	SELKEY	Handle Device Select (GOLD Key)
24021	TSTCTU	Execute Tape Test
24024	CONDTN	Condition Tape
24027	RSTCTU	Soft Reset Tapes
24032	IOCNTL	Set Up for I/O Control Escape Sequence (ESC &p)
24035	IOSTGO	Send I/O Device Status to Data Comm
24040	IODNGO	Send I/O Control Completion Code to Data Comm
24043	IORDGO	Send I/O Record(s) to Data Comm
24046	RCRDGO	Enter Record Mode
24051	BNRYGO	Send Binary I/O Data to Data Comm
24054	CTDCDP	Fast Binary Read (ESC e)
24057	CTMON	Monitor Cartridge Tapes
24062	PTTPLN	Record Top Line of Display
24065	TIDOO	Initial Tape Interrupt Vector Address
24067	RDABRT	Abort File Read
24072	BSYCHK	Check for Busy Tape Drives
24075	CTINTR	Cartridge Tape Interrupt Routine

Local Variables

Local variables for the I/O subroutines are stored in two regions of RAM: 71 bytes at 177440-177546 (octal) and 24 bytes at 177150-177177 (octal). Many of the variables used by the I/O module are also accessed by the Main Code module. Any changes in the definition of the I/O variables should be coordinated with corresponding changes in the Main Code module as required.

Fast Access RAM

No fast access RAM is allocated for the I/O subroutines other than the stack.

FUNCTIONAL DESCRIPTION

The I/O code is responsible for handling all commands involving the tapes and printer (and alternate I/O device, if installed). The code may be roughly divided into the command interpreters, the logical (i.e., device independent) drivers, and the hardware drivers for each device.

Location IOCERR is used to flag the return condition for various routines. The flag consists of either an ASCII S (123B), F (106B), or U (125B). The letter S represents a successful return; F represents a fail return; and U, a return due to a user interrupt (user pressing RETURN key on keyboard to abort operation).

Command Interpreters

I/O Control Escape Sequence (ESC & p): IOCNTL

Characters sent to the display are interpreted in the main code routine CHINTO by table lookup (see main code functional description). A pointer indicates which table is to be used. Receipt of the character string escape, ampersand (&), lower-case p, causes control to be passed to the I/O routine, which changes the table pointer so that subsequent characters will be interpreted according to the table IOCTAB in the I/O code.

Plus (+), minus (-), and numeric characters cause control to be passed to the main code routines DCPLUS, DCMNUS, and DCNUM, respectively. Other characters that are valid in an I/O control escape sequence (B, b, C, c, D, d, F, f, M, m, P, p, R, r, S, s, U, u, W, w, ^, ~, and space) are interpreted by the routines IOC010 through IOC120. Other characters terminate the escape sequence.

IOC020 through IOC120 all exit to IOCEX0, which checks whether the character just received was capitalized (^ is treated as capital ~). If so, control is passed to the logical driver which implements the required command. (Control is passed via the transfer table IOCMTB). If the character is not capitalized, control is returned to the main code to await receipt of the next character.

Green Key: IOCKEY

Whenever the green key is pressed, control passes to the IOCKEY routine. This routine monitors the keyboard until a valid command key or RETURN is pressed. Control is then passed (via an address from GRNTBL) to another routine.

If the command does not require further keyboard input (e.g., the copy commands or read beyond end of data) control is passed directly to the logical or hardware driver that implements the required function.

Commands requiring only a device specification (rewind and mark file) pass control to USRNPM. This routine monitors the keyboard for a device key, and when one is pressed, passes control to logical driver CTRLIO.

Commands requiring a parameter (skip lines and find file) cause control to transfer to USS010, which accumulates the parameter and checks for a device key. When a device key is pressed, control is passed to the logical driver CTRLIO.

Condition (Control TEST): CONDTN

CONDTN is called from the main code whenever control TEST is pressed. CONDTN jumps to USPNPM, which monitors the keyboard for a device selection. When a device key is pressed, control is passed to the logical driver CTRLIO.

Gold Key: SELKEY

SELKEY implements the gold key (device selection) function. It is described in the section on entries to the I/O code, below.

Logical Drivers

Copy, Compare: XFRD2D

Data transfers to and from devices are record-oriented. Input from devices is accomplished via calls to GETIO, which passes control to the driver for the selected device via transfer table D2BFTB. Similarly, output to devices is accomplished via calls to PUTIO, which transfers via BF2DTB to drivers for all selected output devices. A description of buffer handling is included in the section on Alternate I/O.

XFRD2D performs all copy and compare operations, whether requested from the keyboard or via escape sequence. For copy operations, XFRD2D alternates calls to GETIO and PUTIO until the transfer limit (line, file, or end of data) is reached. For compare operations, XFRD2D calls GETIO once for each of the devices, and then calls CMPBFS to compare the records. If the records match, the process is repeated until the compare limit is reached.

Control: CTRLIO

CTRLIO handles all I/O control operations that do not involve data transfer. (There are a few exceptions; e.g., the tape monitor CTMON rewinds newly-inserted tapes without invoking CTRLIO, and the read- beyond-end-of-data routine EVDRED is called directly from IOCKEY.)

The calling routine passes three pieces of information to CTRLIO: flags for the device(s) on which the operation is to be performed (stored in IOCDEV), a number identifying the operation to be performed (IOCTYP), and a parameter, if required (sign in IOPSGN, magnitude in IOCCNT). CTRLIO calls the driver for each device specified in IOCDEV by transferring through table CTLTAB. The hardware driver for each device interprets the operation type and parameter.

Escape Sequence Read (ESC & p R): IOREAD, IORDGO, BNRYGO

This routine is called by the escape sequence routine (see IOCNTL, above) only if the escape sequence came from the data comm. IOREAD sets a flag (SDVREC) in MFLGS and returns to the main code. This bit informs the main code that the I/O routine IORDGO should be called when a data comm handshake has been completed.

IORDGO reads a record by calling GETIO. If an ASCII read was requested, the record is output by calls to EXPBUF. This utility routine expands display codes to escape sequences and calls the main code routine XPUTDC to output each character.

If a binary read is requested, IORDGO reads the record and passes control to SDBYCT. This routine sends the byte count to the data comm, sets the SBINRY bit in MFLGS and returns to the main code. The SBINRY bit causes the main code to call the I/O routine BNRYGO when a second handshake is completed. BNRYGO sends the binary data to the data comm via calls to XPUTDC.

If a file read is requested, IOREAD sets the FILRED flag in IOFLGS. IORDGO and BNRYGO check this flag after sending a data record; if it is set, the routines set SDVREC before exiting to trigger the next record transfer after a handshake.

Read Key: REDKEY, CTLRED

REDKEY and CTLRED implement the READ key and control READ functions, respectively. These functions request a local read to display, read to data comm with handshake, or read to data comm without handshake, depending on the terminal mode and strapping (see Reference Manual and descriptions of REDKEY, CTLRED, below for complete description). In each case, one file is read from the current "FROM" device.

Local read is implemented by a call to XFRD2D.

Read to data comm is implemented by repeated calls to IORDGO. Two flags, both in IOFLGS, control the read. RDWOWT is set for a read without handshake, and causes IORDGO to loop until a file mark is read. USREAD causes IORDGO to set the SDVREC bit in MFLGS after sending each data record. SDVREC causes the main code to call IORDGO after a handshake has been completed.

Escape Sequence Write (ESC & p W): IOWRIT

This routine is called by the escape sequence routine (see IOCNTL, above) only if the escape sequence came from data comm.

This routine implements both ASCII and binary write commands. ASCII records are read from the data comm by the DC2BUF routine; binary records are read in a loop in IOWRIT (address IOW025). The record is sent to all "TO" devices by a call to PUTIO.

Record Key: RECKEY, RCRDGO

RECKEY decides what to do when the RECORD key is pressed. If the terminal is in Record or Edit Mode, RECKEY jumps to USREDA (see below). Otherwise, if the terminal is in Local Mode, RECKEY copies the display to all "TO" devices via a call to XFRD2D.

If the terminal is in Remote Mode (and not doing Data Logging), RECKEY enables Record Mode by setting the RECORD flag in MDFLG1. When Record Mode is triggered (character other than CR or LF received from data comm), the main code calls RCRDGO. RCRDGO reads records from the data comm (via DC2BUF) and copies them to all "TO" devices (via PUTIO) until an error occurs or the RECORD key is pressed again. In this case, the RECORD key is handled by RCRDGO instead of RECKEY.

Status Request (ESC & p ^): IOSTAT, IOSTGO

IOSTAT sets the SDVST flag in MFLGS to signal the main code that device status should be sent after a data comm handshake. IOSTAT puts the device number of the device for which status is being obtained in IOSTA0. Then IOSTAT jumps (via transfer table STATTB) to the routine which generates status for the selected device. The device routines put the status in RAM locations IOSTA1 to IOSTA3. After a handshake, the main code calls IOSTGO to output the status.

Edit, Data Logging: USRTED, PTTPLN, USREDA

IOCKEY jumps to USRTED whenever the f4 function is pressed after the GREEN key. USRTED turns Edit Mode (or Data Logging Mode) on or off by setting or clearing the EDIT flag in MDFLG1.

When the terminal is in Edit or Data Logging Mode, and a line of text is about to roll off the top of the screen, the main code calls PTTPLN. PTTPLN reads the top line from the display and writes it on all "TO" devices by calling PUTIO.

USREDA implements the RECORD key function for Edit and Data Logging Modes. This routine copies the display to all "TO" devices by calling PTTPLN through the main code routine PTBLKO. If the terminal is in Data Logging Mode, USREDA then exits. If in Edit Mode, USREDA copies the last line to the "TO" devices by calling PTTPLN directly, then copies the rest of the input file to the "TO" devices by calling XFRD2D, and finally exits Edit Mode by clearing the EDIT flag.

Fast Binary Read (ESC e): CTDCDP

CTDCDP reads records from the "FROM" device by calling GETIO. The records are output via calls to BNRYGO.

Hardware Drivers

CTU General

The cartridge tape hardware is multiplexed--only the selected unit may move or generate interrupts. The hardware drivers for the left and right tape units select the required unit (via SELLCT or SELRCT) by appropriately setting the unit select bit in the CTU command word and copying the information for that unit into the RAM locations between UNIT0 and SFTCNT, inclusive (information for the unit not selected is saved in the RAM area OTHER). These hardware drivers then branch to the unit-independent drivers discussed below.

CTU Interrupt Routine: CTINTR

Four conditions cause tape interrupts: (1) tach edge (TAK bit in tape status goes high), (2) hole detected (HOL bit goes high), (3) byte read from tape (RDY bit goes high while reading), (4) unit ready to write byte on tape (RDY bit goes high while recording). The interrupt routines determine the cause(s) of the interrupt by examining the tape status.

The CTU interrupts to location 50 octal in the main code. The main code determines whether the I/O code is installed; if so, it jumps to CTINTR. CTINTR is responsible for monitoring holes and setting UNIT0 accordingly, and for maintaining an absolute tach counter for the selected unit (used by the GREEN, SPACE routine). After handling these, CTINTR jumps through CTIVEC to a subroutine for the particular tape function being performed. (Note: for easy identification, these subroutines all begin with the letters "TI", as TID00, TIEBK, etc.) It is the responsibility of the hardware drivers to correctly set up CTIVEC before moving the tape.

CTU Input: CT2BUF

CT2BUF causes a record to be read from the selected tape and returns a pointer to the buffer containing the record. Basically, CT2BUF sets up CTIVEC, starts the tape moving, and monitors the two I/O buffers until the interrupt routine marks one as ready. See the section on alternate I/O for a description of use of the I/O buffers.

To improve throughput by keeping the tape running, the tape reading interrupt routine TIGCT1 may initiate another read after a record has been read. In this case, the tape will be moving when CT2BUF is called. CT2BUF will just monitor the buffers until one is ready.

CTU Output: BUF2CT

BUF2CT causes a record to be written on the selected unit. In the process of initiating a write, BUF2CT clears the buffer status bit for the selected tape and sets the high-order status bit to hold the buffer until the recording is finished. Once the recording operation has been initiated, BUF2CT returns, allowing the interrupt routines to complete the operation.

Upon completing a record, the recording interrupt routine TIPCT1 checks the status of the other buffer to see whether it is designated for output to the selected tape. If so, TIPCT1 initiates recording of that buffer. BUF2CT detects this situation by checking the buffer status when called.

CTU Control: CTLCT

CTLCT takes as a parameter a number identifying the required operation in RAM location IOCTYP (see CTRLIO, above). CTLCT then uses this number to branch to the correct tape driver through transfer table CTCTLT.

Printer Output: BF2PRT

The 2645 firmware supports three printer interfaces: parallel, RS-232 serial, and video output. The type of interface card (if any) installed in the terminal is determined by the main code on power-up and reset. This information is stored in the RAM location PTRFLG.

BF2PRT calls PTRCHR to output each character. PTRCHR checks PTRFLG to determine which driver to call: PRCHR1 for the parallel interface and video output, PRCHR2 for the serial RS-232.

BF2PRT is also responsible for interpreting spacing information encoded in records read from a formatted display. Each physical record read from a formatted display corresponds to one line of the display containing at least one unprotected field. Every such record begins with an octal 304 byte (identifying it as a Format Mode record), followed by a byte giving the number of lines in the display between that line and the last previous line containing unprotected fields. In addition, every field in the record is preceded by a 304 byte followed by the number of spaces between that field and the previous field (or beginning of the line, if it is the first field). BF2PRT translates this information into an appropriate number of line feeds and spaces so that the printed fields will be spaced the same as the formatted display.

Printer Control: CTLPRT

Only two printer control commands are implemented: skip lines and form feed (any unrecognized control request is translated into a form feed). CTLPRT executes both commands by calling PTRCHR.

Display Input: DSP2BF, PTTPLN

DSP2BF is called by GETIO, and thus is involved in all copy and read operations which use the display as input. Characters are read from the display via calls to the main code routine GETDSP. DSP2BF is responsible for inserting the spacing information in Format Mode records mentioned above.

PTTPLN is called in Edit Mode to output lines as they roll off the top of display memory. This routine gets characters from the display via the main code routine NXTCHR. NXTCHR differs from GETDSP in that the former does not involve the cursor. Since Edit Mode and Format Mode are incompatible, PTTPLN is not concerned with Format Mode records.

Display Output: BF2DSP

BF2DSP sends characters to the display via the main code routine CHINT.

Alternate I/O: BF2ALT, ALT2BF, CTLALT, STALT

Each of these routines passes the address of an alternate I/O vector to the main code routine IORMGO. IORMGO determines whether an alternate I/O ROM is installed, and if so, jumps to the vector. See the section on Alternate I/O for more details.

SUBROUTINE SPECIFICATIONS

Each subroutine will use the registers as specified below. Any registers not specified in the exit list are returned with their contents undisturbed. The settings of the processor flags (S, Z, P, and C) are generally not retained. Certain routines use the processor flags to represent exit conditions. These conditions are listed in the subroutine headers.

Execute GREEN Key Functions

```
IOCKEY - PROCESS KEYBOARD I/O CONTROL SEQUENCES

  ENTRY:  DON'T CARE

  EXIT :  ALL REGISTERS DESTROYED
          NC => NO ERROR
            IOCERR = S
          C => ERROR
            IOCERR = U => USER INTERRUPT
            IOCERR = F => FAILURE
```

This routine receives control whenever the GREEN key is pressed. It processes all subsequent keyboard input until:

(1) The user aborts the keyboard sequence by pressing RETURN.

(2) An unattended operation is specified and successfully started.

(3) An attended operation is specified and successfully completed.

(4) The user aborts an attended operation by pressing RETURN.

(5) An attended operation is interrupted by an error and the user responds to the error message.

(6) An error prevents the successful start of a specified unattended operation, and the user responds to the error message.

Handle READ Key

```
      REDKEY - USER PRESSED "READ" KEY

        ENTRY:  DON'T CARE

        EXIT :  LOCAL OR BLOCK MODE - LOCAL READ FINISHED
                REMOTE, CHARACTER MODE - MFLGS2 [SDVREC] = 1
                ALL REGISTERS DESTROYED
```

The operation of this subroutine depends on the REMOTE and BLOCK MODE latching keys.

Local Read (REMOTE up or BLOCK MODE down) - the subroutine executes a local read to the display until:

(1) Successful completion of the read (end of file).

(2) Interruption by the user (RETURN).

(3) An error is detected and the user responds to the error message.

Remote Read (REMOTE down and BLOCK MODE up) - the subroutine sets up a remote read by setting IOFLGS [USREAD] and MFLGS2 [SDVREC].

Handle CONTROL READ

```
      CTLRED - USER PRESSED CONTROL READ

        ENTRY:  DON'T CARE

        EXIT :  LOCAL - LOCAL READ FINISHED
                REMOTE, BLOCK, LINE - MFLGS2 [SDVREC] = 1
                OTHER - REMOTE FILE READ FINISHED
                ALL REGISTERS DESTROYED
```

This subroutine handles the Control-Read function. In Local Mode, CTLRED performs a local read in the same manner as REDKEY (q.v.). In Remote, Block, Line Mode, CTLRED performs the same function as REDKEY in Remote, Character Mode. In Remote, Character Mode or Remote, Block, Page Mode, CTLRED calls IORDGO (q.v.) until a file is read or an error detected.

Handle RECORD Key

```
      RECKEY - USER PRESSED RECORD KEY

        ENTRY:  DON'T CARE

        EXIT :  RECORD MODE ENABLED => RECORD MODE EXITED
                EDIT MODE => EDIT MODE TERMINATED
                LOGGING MODE => DISPLAY COPIED TO "TO" DEVICES
                LOCAL => DISPLAY COPIED TO "TO" DEVICES
                REMOTE => RECORD MODE ENABLED
                ALL REGISTERS DESTROYED
```

This subroutine handles all of the functions of the RECORD key except for turning off Record Mode once it is triggered.

Note that Record Mode is only enabled by this routine. The Main Code detects the triggering condition (character other than CR or LF received from data comm) and enters Record Mode by calling RCRDGO (q.v.).

Handle Device Select (GOLD Key)

```
      SELKEY - SELECT "FROM" AND "TO" DEVICES

        ENTRY:  DON'T CARE

        EXIT :  ERROR => NO CHANGE IN DEVICE ASSIGNMENTS
                NO ERROR => DEVICES RESELECTED
                ALL REGISTERS DESTROYED
```

This subroutine handles subsequent keyboard input as follows:

(1) GOLD Key: Abort device selection. No change in "from" and "to" devices.

(2) f1 - f8: Accumulates the selected device(s). Key f4 is invalid, and is ignored.

(3) Other:

   (a) Valid selection (at most one "from" device): The "from" and "to" devices are changed, if any were selected. A call is made to the keyboard code module to cause the last key hit to be reissued on the next call to "ZGTKEY".

   (b) Invalid selection (more than one "from" device): An error message is displayed. SELKEY returns when the user presses RETURN. The last key is not executed. Device selections are not changed.

Tape Test

```
      TSTCTU - DO COMPLETE TEST OF TERMINAL INCLUDING
        TEST OF BOTH TAPE UNITS.

        ENTRY:  DON'T CARE

        EXIT :  SUCCESSFUL TEST => RETURN AFTER LAST SELF-TEST
                ERROR => DISPLAY ERROR MESSAGE AND HANG TERMINAL
                ALL REGISTERS DESTROYED
```

This routine performs the keyboard initiated tape test procedure:

(1) A tape test is performed on the left drive.

(2) Two terminal self-tests (TRMTST) are performed.

(3) A tape test is performed on the right drive.

(4) Another terminal self-test is performed.

This procedure provides the proper duty cycle on the tape drives for the "burn-in" process during manufacturing of the terminal.

If any errors occur, an error message is displayed and the terminal is made to "hang" by going to "HANGU0" in the main code module. Control is returned to the calling routine only if the entire procedure is completed successfully.

Condition Tape

```
      CONDTN - PROCESS CONDITION TAPE COMMAND FROM KEYBOARD

        ENTRY:  DON'T CARE

        EXIT :  ALL REGISTERS DESTROYED
```

This routine processes the condition tape command from the keyboard. The keyboard is monitored for another key hit. If a device specifier key is hit (f1-f8), the control function - ESC & p 4 C - is executed on the selected device. If the device is a cartridge tape, a tape condition operation is performed on the specified drive. Control is returned to the calling routine after the control operation is completed. If the RETURN key is hit, a return is made immediately without any control operation being performed. Any other key hit causes the bell to be sounded.

Soft Reset Tapes

```
RSTCTU - CTU SOFT RESET

  ENTRY:  INTERRUPTS DISABLED

  EXIT :  INTERRUPTS ENABLED
          ALL REGISTERS DESTROYED
```

This routine is called whenever a soft reset is executed. If a tape is moving when this routine is called, the tape is rewound to the Load Point (if a record operation is in progress, a File Mark and End of Data Mark are written before the rewind). No action is taken for tapes that are not moving. I/O buffers are freed.

Set Up for I/O Control Escape Sequence

```
IOCNTL - <ESC><&><LOWER CASE P> RECEIVED

  ENTRY:  DON'T CARE

  EXIT :  RNGTA = IOCTAB
          ESCFLG = 2
          I/O CONTROL VARIABLES CLEARED
          A,C,H,L DESTROYED
```

This routine is called whenever the terminal receives the I/O control escape sequence header (ESC & p) from any source. This routine sets up the terminal to interpret the parameters of the escape sequence by clearing the I/O control variables and setting the character interpretation table (RNGTA) to the table of I/O control routine vectors (IOCTAB). Setting ESCFLG to two prevents the main code from resetting RNGTA.

Send Device Status to Data Comm

```
IOSTGO - SEND DEVICE STATUS

  ENTRY:  IOSTA0 = DEVICE CODE
          IOSTA1-IOSTA3 = DEVICE STATUS
          HANDSHAKE COMPLETED

  EXIT :  STATUS SENT
          A,B,C,H,L DESTROYED
```

This routine transmits the information requested by a device status escape sequence (ESC &p^). The status information will have already been gathered by an I/O control routine entered from the main code via IOCTAB (see IOCNTL). The control routine sets a bit (MFLGS [SDVST]) that signals the main code to call

IOSTGO when a data comm handshake has been completed. IOSTGO sends the status information out the data comm as an escape sequence: ESC \ p <device code> <status byte 0> <status byte 1> <status byte 2>. The device status pending flag (MFLGS [SDVST]) is cleared after the device status is transmitted.

Send I/O Control Completion Code to Data Comm

```
IODNGO - SEND OPERATION COMPLETED RESPONSE

ENTRY:  IOCDPT = COMPLETION TYPE (S, F, OR U)
        HANDSHAKE COMPLETED

EXIT :  COMPLETION CODE SENT
        A,B,C,H,L DESTROYED
```

Most I/O control escape sequences return completion codes when received from the data comm. The routines which execute these control functions put the completion code (S = success, F = fail, U = user interrupted) into IOCDPT and set the device completion code transfer pending bit (MFLGS [SDVDUN]). When a data comm handshake is completed, this routine is called to transmit the completion code. After the completion code has been transmitted, the transfer pending bit is cleared.

Send I/O Record(s) to Data Comm

```
IORDGO - TRANSFER RECORD TO DATA COMM

  ENTRY:  HANDSHAKE COMPLETED
          IOCTYP = TRANSMISSION TYPE
                 0 = ASCII, NEXT BLOCK
                 1 = ASCII, LAST BLOCK
                 2 = BINARY, NEXT BLOCK
                 3 = BINARY, LAST BLOCK
                 4 = ASCII FILE READ
                 6 = BINARY FILE READ
          IOFLGS [USREAD OR FILRED] = 1 => READ FILE
          IOFLGS [RDWOWT] = 1 => READ FILE W/O HANDSHAKE
          LSTRED -> START OF LAST BLOCK
               (0 => NO LAST BLOCK)
          NXTRED -> START OF NEXT BLOCK
               (LSB=0 => GET NEW BUFFER FULL)
          NOTE: ASCII TRANSFER IS 1 FIELD (FORMAT
                 RECORD) OR 1 NORMAL RECORD.
                 BINARY TRANSFER IS ALWAYS 1 RECORD

  EXIT :  ALL REGISTERS DESTROYED
          LSTRED, NXTRED UPDATED IF NEXT BLK
               WAS REQUESTED
          ASCII TRANSFER - BLOCK SENT
          BINARY TRANSFER - BYTE COUNT SENT,
               MFLGS2 [SBINRY] = 1
          FILE READ, READ W/O WAIT - FILE SENT
          FILE READ, READ WITH HANDSHAKE - RECORD SENT,
               MFLGS2 [SDVREC] = 1
```

In any read requiring a data comm handshake (e.g., READ key or ESC & p <read byte control> R), the initial handling routine sets the device record transfer pending flag MFLGS [SDVREC] (see REDKEY). After a data comm handshake is completed, this routine is called to perform the required transfer operation.

If an ASCII transfer is specified, IORDGO transmits the data directly. Otherwise, for a binary read, IORDGO sends the byte count and sets MFLGS2 [SBINRY], which signals the main code to call BNRYGO (q.v.) to send the binary record when the next data comm handshake is complete. In either case, the transfer pending flag MFLGS [SDVREC] is cleared.

Enter Record Mode

```
      RCRDGO - ENTER RECORD MODE

        ENTRY:  RECORD MODE ENABLED (VIA "RECORD" KEY)
                RECORD MODE TRIGGERED (VIA RECEIPT OF CHARACTER
                     FROM DATA COMM OTHER THAN CR OR LF)

        EXIT :  ALL REGISTERS DESTROYED
                RECORD MODE EXITED BECAUSE OF ERROR OR USER INTERRUPT
```

The user enables Record Mode by pressing the RECORD key (see RECKEY). When the main code detects a character from the data comm that is not a CR or LF, it calls RCRDGO to enter Record Mode. RCRDGO copies data from the data comm to the "to" device(s) until there is an error or until the user terminates Record Mode by pressing the RECORD key again. If terminated by an error, RCRDGO does not return control until the user responds to the error message.

Send Binary I/O Data to Data Comm

```
      BNRYGO - SEND BINARY RECORD

        ENTRY:  LSTRED -> FIRST BYTE OF RECORD

        EXIT :  ALL REGISTERS DESTROYED
                RECORD SENT
                BUFFER RELEASED
```

BNRYGO is called to do a binary record transmit after IORDGO (q.v.) has sent the byte count and a second data comm handshake is completed.

BNRYGO exits in one of four ways. If an error occurs, BNRYGO aborts the binary read via RDABRT (q.v.). If there were no errors and the request was for one record (ESC & p 2 R), BNRYGO returns to the main code. If the request was for a file (ESC &p 6 R), BNRYGO first sets MFLGS2 [SDVREC] so that after the next data comm handshake the main code will call IORDGO to start sending the next record. If a read without data comm handshake is in progress (ESC &p 6 R, Block, Page Mode), BNRYGO jumps directly to IORDGO instead of setting MFLGS2 [SDVREC].

Fast Binary Read (ESC e)

```
       CTDCDP - FAST BINARY READ

         ENTRY:  DON'T CARE

         EXIT :  ALL REGISTERS DESTROYED
```

This routine is called whenever the terminal receives an ESC e from any source. This routine transmits a file of binary data to the data comm. The routine returns after the file is transmitted or when an error occurs.

Monitor Cartridge Tapes

```
       CTMON - MONITOR CARTRIDGE TAPE UNITS

         ENTRY:  DON'T CARE

         EXIT :  A,H,L DESTROYED
```

This subroutine looks for tapes that have just been inserted or removed. Newly inserted tapes are rewound to the load point if and only if the tape units are not busy (i.e., the RUN bit is off in the last command issued). Otherwise, the new tape is ignored until the next call to CTMON. Eject lights are turned off for tapes that have been removed.

CTMON is called by all wait loops to insure that tapes cannot be removed without the terminal knowing about it.

Record Top Line of Display

```
       PTTPLN - OUTPUT TOP LINE OF DISPLAY MEMORY

         ENTRY:  D,E -> (1ST CHAR IN LINE)+1

         EXIT :  C => ERROR, LINE NOT OUTPUT
                 NC => NO ERROR, LINE SENT TO ALL "TO" DEVICES
                 A,B,H,L DESTROYED
```

This subroutine is called to roll a line off the top of display memory in Edit and Data Logging Modes.

In the event of an error, PTTPLN returns (with Carry set) after the user responds to the error message.

Initial Tape Interrupt Vector

```
      TIDOO - "DO NOTHING" TAPE INTERRUPT HANDLER
```

TIDOO is an address, rather than an entry vector. It is used by the main code to initialize a vector (CTIJMP) used by the tape interrupt routine.

Abort User Read

```
      RDABRT - ABORT USER READ OPERATION

        ENTRY:  USER READ IN PROGRESS
                  (IOFLGS [USREAD] = 1 OR IOFLGS [FILRED] = 1)

        EXIT :  READ ABORTED
                BUFFERS FREED
                A,B,C,H,L DESTROYED
```

This routine is called by the main code if the user presses RETURN while the terminal is waiting for a data comm handshake required by a file read operation.

Check for Busy Tape Drives

```
      BSYCHK - CHECK IF CTU BUSY

      THIS ROUTINE WAITS UNTIL CTU NOT BUSY OR USER  INTERRUPT.  DISPLAYS "CTU
      BUSY"  MESSAGE  TAPES  INSERTED  DURING THE WAIT ARE REWOUND BEFORE BSYCHK
      RETURNS.

            ENTRY:  DON'T CARE

            EXIT :  NC => CTU NOT BUSY
                    C => USER INTERRUPTED
                    A,H,L DESTROYED
```

This subroutine is called before terminal self-test is performed to permit any unattended CTU operations to complete before interrupts are disabled for the test.

If the CTU is not busy (RUN off in last command), BSYCHK returns NC immediately. If busy, BSYCHK displays a "BUSY" message until the CTU is not busy or until the user interrupts the operation by pressing RETURN.

Cartridge Tape Interrupt Routine

```
      CTINTR - PROCESS CARTRIDGE TAPE INTERRUPTS

        ENTRY:  RETURN ADDRESS,PSW,H,L PUSHED ONTO STACK
                INTERRUPTS DISABLED

        EXIT :  RETURN FROM INTERRUPT
                FLAGS, REGISTERS RESTORED
                INTERRUPIS ENABLED
```

This routine handles all cartridge tape interrupts. The main code saves the PSW and H and L registers, and does a poll to determine which device interrupted.

CTINTR re-enables interrupts before returning. If an interrupt requires much processing (e.g., to start writing another record after one record is completed), CTINTR re-enables interrupts before returning (interrupts should not be disabled for more than 300 microseconds). Therefore, more than one call to CTINTR may be on the stack concurrently. Note that CTINTR starts immediately at location 24075B rather than having a vector. This helps to streamline interrupt handling.

HP 2645A KEYBOARD CODE INTERFACE

INTRODUCTION

The keyboard code module handles all keyboard subsystem functions. The module detects key hits, sets the lights on the keyboard, and maintains the current state of the latching keys and keyboard interface option switches. Additionally, the module includes routines to perform the alpha and numeric field checking operations of the terminal. This allows changes in character attributes for foreign keyboards to be reflected in the type check routines.

The keyboard code occupies the 2K region from 18K to 20K (44000 to 47777 octal) in the ROM space.

INTERFACE SPECIFICATIONS

Entry Vectors

The entry vectors to the keyboard routines begin at location 44002 (octal). The vectors consist of "jumps" (JMP) to the corresponding subroutine starting locations:

Location	Name	Function
044002	INITKB	Initialization Routine
044005	GTKEY	Get Keyboard Input
044010	KBCTL	Keyboard Control
044013	KBMON	Monitor Keyboard
044016	SETMD1	Set Terminal Mode 1 Flags
044021	CLRMD1	Clear Terminal Mode 1 Flags
044024	BELL	Sound the Keyboard Bell
044027	SETXMT	Set the Transmit Light
044032	CLRXMT	Clear the Transmit Light
044035	STJMPR	Set Jumper Escape Sequence Processor
044040	STLKYS	Set Latching Keys Escape Sequence Processor
044043	ALPCHK	Alpha Field Check Routine
044046	NUMCHK	Numeric Field Check Routine

Local Variables

Local variables for the keyboard subroutines are stored in the RAM region 177400-177437 (octal). The definition of this area is at the discretion of the particular keyboard module used.

Fast Access RAM

The keyboard subroutines may use sixty-four (64) bytes in the fast access RAM region at locations 110700-110777 (octal).

Keyboard Constants

The following constants are to be included in the keyboard code address space:

Location	Usage
044051	Initial alternate character set
044052	Initial alternate character set for output

Locations 44051-44052 may be set to either 0, 20, 40, or 60 octal corresponding to the base character set and alternate character sets 1, 2, or 3.

Location 44051 contains the alternate character set code to be used if a Shift Out (SO) control code is received by the terminal when no alternate character set defining escape sequence has been received before.

Location 44052 contains the initial active alternate character set to be used when tearing apart the screen for transmission to the host computer or for storage on an I/O device. Normally, this value is zero, but is set to the foreign language set in foreign terminals to inhibit the generation of an alternate character set defining escape sequence (ESC)) when the display shifts into the foreign character set.

FUNCTIONAL DESCRIPTION

General

The keyboard module is organized to provide a flexible structure to facilitate the development of foreign or alternate keyboards. The keyboard module may be divided into the following sections: monitor, keyboard input processor, and utilities.

Monitor

The keyboard monitor routine scans the keyboard for changes in the states of the keys on the keyboard. The keys on the keyboard are organized in a matrix of 14 columns of eight keys. The current state of the keyboard is maintained by the monitor routine in a 14-byte bit table. Each key on the keyboard corresponds to one bit in the bit table. A scan of two columns in the keyboard matrix is made on each call to the monitor routine. If any key transitions are detected, an entry is made into a transition buffer and the current keyboard state table is updated. Then the monitor routine is terminated.

Each entry in the transition buffer consist of two bytes: the column number in which the transition occurred and the new state for the column. These entries are later interpreted by the keyboard input processor. An entry in the buffer may actually contain multiple key transitions for a key column. The transition buffer contains space for up to 20 transitions.

The monitor routine is called each time the processor's timer interrupts (about every 10 milliseconds). This results in a minimum interval of 70 milliseconds for a full keyboard scan when no key transitions occur. Additional scans are made on calls to the keyboard input processor to provide a higher scan rate.

Keyboard Input Processor

The keyboard input processor (GTKEY) takes the input from the transition buffer and performs the proper action for the key transition(s). The action may be to return a character code to the calling routine or to perform an internal operation.

Another state table, identical to the state table maintained by the monitor, is maintained by the keyboard input processor. The alternate state table reflects the transitions that have been processed by the keyboard input processor.

When a key is pressed down, the action taken is derived from an index table. Separate tables are defined for actions to be taken when the shift keys are up and when either shift key is down. This corresponds to the lower and upper case sets. The table index is computed by multiplying the column number of the key by 8 and adding the number of the bit that corresponds to the key.

Each entry in the index table consist of one byte. If the high order bit of the entry is zero (0), the entry represents an ASCII character to be returned to the calling routine. These types of entries are used for the character set and numeric parts of the keyboard. Entries in the range 200-207 (octal) represent actions to be performed internal to the keyboard module. These types of entries are used for the latching keys and keys which switch functions on and off (e.g., MEMORY LOCK). The action may result in no data returned to the calling routine or a value in the range 260-377 (octal) returned. Entries in the range 230-237 (octal) represent keys that perform internal terminal functions (e.g., READ and RECORD). Entries in the range 260-377 (octal) represent keys that generate escape sequences (e.g., ROLL UP). In general, the escape sequences are two-character escape sequences where the second character of the escape sequence is derived by masking out the high order bit of the entry. Some codes are used to generate three or more character escape sequences. In particular, the 377 (octal) code is used to signify the display enhancement code (ESC & d). The last code returned from a call to GTKEY is stored in location KBCHAR.

All keys, except the latching and internal terminal function keys, repeat if the key is held down. Repeating is accomplished by setting a repeat counter, KBTIMR, to the number of 10 millisecond intervals plus one before the key is to be repeated. The monitor routine decrements the repeat counter to the value of one. If no new keys are pressed down when the input processor is called and the repeat counter contains a value of one, the last key hit is repeated. The code for the last key hit is reissued from location KBCHAR and the repeat counter is reset for another repeat interval. Three repeat intervals are used: a start repeat delay for the cursor control keys, a start repeat delay for all other keys, and a repeat delay after repeating has begun. The repeat counter is set to zero (0) when no key is repeating.

Generally, the only action taken when a key is released is to update the alternate state table and to clear the repeat counter if the key released was repeating. Only the latching keys require additional action when the key is released. The action consist of clearing a corresponding bit in location MDFLG2.

Utilities

A number of utility routines are included in the keyboard module. These involve the setting and clearing of the lights on the keyboard, processing the escape sequences to set and clear the latching keys and keyboard option switches, sounding the bell, and performing the type checks for alphabetic and numeric characters.

SUBROUTINE SPECIFICATIONS

The subroutine descriptions will specify the parameters to be included in th registers on entry and the register results on exit. Any registers no specified in the exit list retain their entry values. The processor flags (S Z, P, and C) are generally altered by the subroutines. Certain routines use the processor flags to signify different return conditions. These conditions are specified in the subroutine descriptions.

The names of the subroutines to be used within the keyboard code block are included in the following specifications. The names within parentheses are the names that will be used by the external code blocks to reference the associated subroutines.

Initialization Routine

```
     INITKB (ZINIKB) - INITIALIZE KEYBOARD

       ENTRY:  DON'T CARE

       EXIT :   A DESTROYED
                NC => NO ERRORS
                C => ERROR DETECTED
                  B,C -> ERROR MESSAGE
```

This subroutine is called whenever power is turned on or a complete terminal reset is being performed. The interrupt system must be disabled by the calling routine when this subroutine is called.

This subroutine initializes the option switch flags (KBJMPR, KBJMP2, and KBJMP3) to the settings of the option switches on the keyboard interface, clears the keyboard state tables (KBBUF and KBBUF2), and turns off all the lights on the keyboard (KBLEDS = 0).

Get Keyboard Input

```
GTKEY (ZGETKY) - GET KEYBOARD INPUT

  ENTRY:  DON'T CARE

  EXIT :  Z => KEYBOARD INPUT PRESENT
            A = KEYBOARD INPUT CODE
          NZ => NO KEYBOARD INPUT
            A = 0 => NO KEY HIT (OR NULL CHARACTER)
            A # 0 => KEYBOARD LOCKED (VALUE IS KEYBOARD
                     INPUT CODE)
          B,C,D,E DESTROYED
```

This routine is called to get input, if any, from the keyboard. The keyboard input code ranges from 0 to 377 (octal). The values from 0 to 177 (octal) represent an ASCII character corresponding to the key that was hit. The values from 260 to 377 (octal) correspond to keys that translate to two-character escape sequences (e.g., ROLL UP, ROLL DOWN, INSERT LINE, etc.). The second character of the escape sequence is extracted by masking out the high order bit of the value (e.g., 301 -> 101 (A)). The values from 360 to 367 (octal) represent the function keys f1 to f8 respectively. The values 230 to 237 (octal) are associated with the following keys:

Value	Key
230	ENTER
231	BREAK
232	DISPLAY FUNCTIONS OFF
233	I/O CONTROL (GREEN)
234	READ
235	RECORD
236	SELECT (GOLD)
237	CONDITION TAPE (control-TAPE TRMTST)

Monitor Keyboard

```
KBMON (ZKBMON) - MONITOR KEYBOARD

  ENTRY:  DON'T CARE

  EXIT :  ALL FLAGS AND REGISTERS DESTROYED
```

This subroutine will be called approximately once every 10 milliseconds on a timer interrupt. Each time this subroutine is called, two columns of the keyboard are scanned for any changes in key state. By being called on a timer interrupt, a minimum scan rate of 70 milliseconds for the keyboard is provided. The current state of the keyboard lights are also set by the monitoring routine.

Control Routine

KBCTL (ZKBCTL) - KEYBOARD CONTROL

ENTRY: A = CONTROL CODE

EXIT : DETERMINED BY INDIVIDUAL CONTROL ROUTINES
GENERALLY, D-L REGISTERS ARE SAVED AND
A-C DESTROYED

This routine is called to perform various control functions on the keyboard subsystem. The control parameters and registers used are as follows:

Control Parameter   Function and Comments

1   LOKKBD - Lock Keyboard

This routine is called to inhibit the GTKEY routine from returning a key hit status (Z). This effectively locks out the keyboard from the terminal. Note that key values are still returned on calls to GTKEY if any keys are hit (see description to GTKEY). Any keys hit while the keyboard is locked are not retained.

2   UNLKKB - Unlock Keyboard

This routine restores the GTKEY routine to normal operation after a control call to lock the keyboard has been made.

3   RPTKEY - Repeat Last Key Hit

This routine causes the last key code returned by the GTKEY routine to be returned again on the next call to GTKEY.

4   STBLMD - Set Permanent Block Mode

This routine is called at power on or full terminal reset initialization to put the terminal into permanent Block Mode. This forces the BLOCK MODE key in the down state and keyboard option switches G and H into the "open" position. The BLOCK MODE key and option switches are held in these states regardless of their physical state and cannot be altered by any escape sequence.

This control function is invoked for data comm protocols which are strictly Block Mode oriented (e.g., multipoint).

5   STRTST - Set Self-test Start Mode

This routine is called at the beginning of a terminal self-test. This causes all the lights on the keyboard to be lit, sounds the bell, and sets the force full reset flag (CMFLGS(FRCRST)) to cause a full reset if the RESET KEY is pressed while the self-test is in progress.

6 ENDTST - End Self-test

This routine is called at the completion of a terminal self-test. The true state of the keyboard lights is restored and the force full reset flag is cleared.

7 RSETKB - Reset Keyboard

This routine is called when a Soft Reset has been initiated. The keyboard state table and buffer are set to indicate no new keys hit, the current state of the SELECT, RECORD, and DISPLAY FUNCTIONS lights and flags are cleared, all the keyboard lights are lit and the keyboard bell is sounded.

8 CKIOKY - Check for I/O Key Down

This routine returns the state of the I/O key (GREEN key) as follows:

```
Z  => I/O Control Key Up
NZ => I/O Control Key Down
A-E Destroyed
```

9 STPRPT - Stop Key Repeat

This routine inhibits the last key hit from repeating. One use of this routine is from the I/O module to inhibit the RETURN key from repeating when the key is used to abort an I/O operation.

10 CKBRKY - Check for Break Key

This routine checks the physical state of the BREAK key. If the BREAK key is down, the keyboard input buffer is flushed until no more keys are pending. The register and condition flags are returned as follows:

```
Z  => Break Key Not Down
  A Destroyed
NZ => Break Key Down
  A-E Destroyed
```

11 SWCHAR - Switch Character Set

This routine is called whenever a Shift In (SI) or Shift Out (SO) control code is executed by the terminal. This routine is primarily used in bi-lingual terminals to select either the Normal or Foreign Mode of operation.

12   SETFRN - Set Foreign Mode

This routine is called to set the proper operating Mode for bi-lingual terminals (foreign or normal) when a Clear Line operation is performed. The operating Mode is set to the last defined setting for the line. This is important when a Clear Line operation is performed at the display border between a foreign and normal field to insure the proper operating Mode for the next input character.

13   STCHST - Set Bi-lingual Output Mode

This routine is called to send an optional SI or SO control code for bi-lingual terminals when the screen is torn apart for transmission or storage on an I/O device. The flags and registers are returned as follows:

```
NC => No Character to Output
  A Destroyed
C  => SI/SO Character to Output
  A = Character to Output
```

14,15   FRNMD1, FRNMD2 - Set Foreign Mode 1 and 2

These routines are called when either an ESC < or ESC >, respectively, is received by the terminal. The actions taken by the routines are defined by the language being implemented by the keyboard module.

Set Terminal Mode 1 Flags

```
SETMD1 (ZSTMD1) - SET TERMINAL MODE 1 FLAGS

  ENTRY:  A = FLAG BIT TO BE SET
          B = 377B => BLINK ASSOCIATED LED
            = 0 => DON'T BLINK ASSOCIATED LED

  EXIT :  A,C DESTROYED
          ASSOCIATED LED, IF ANY IS SET ON
```

The bits in location MDFLG1 are associated with operating modes of the terminal (see Common Area Allocation). In addition, some of these modes are associated with a light on the keyboard to indicate whether the mode is on or off. This routine is called to set a bit in MDFLG1 and consequently, to set the corresponding light, if any.

Clear Terminal Mode 1 Flag

```
     CLRMD1 (ZCLMD1) - CLEAR TERMINAL MODE 1 FLAGS

       ENTRY:  A = FLAG BIT TO BE CLEARED

       EXIT :  A,C DESTROYED
               ASSOCIATED LED, IF ANY, IS TURNED OFF
```

This routine performs the converse operation of SETMD1. The specified bit and the corresponding keyboard light, if any, are cleared.

Sound the Keyboard Bell

```
     BELL (ZBELL) - SOUND THE KEYBOARD BELL

       ENTRY:  DON'T CARE

       EXIT :  A DESTROYED
               Z FALSE
```

This routine is called to sound the bell on the keyboard.

Set/Clear the Transmit Light

```
     SETXMT (ZSTXMT) - SET TRANSMIT LED

       ENTRY:  DON'T CARE

       EXIT :  A,H,L DESTROYED

     CLRXMT (ZCLXMT) - CLEAR TRANSMIT LED

       ENTRY:  DON'T CARE

       EXIT :  A,H,L DESTROYED
```

These routines set and clear the transmit light in the keyboard.

Set Jumper Escape Sequence Processor

```
     STJMPR (ZSTJPR) - SET KEYBOARD JUMPER ESCAPE SEQUENCE

       ENTRY:  DON'T CARE

       EXIT :  A,H,L DESTROYED
               RNGTA -> FUNCTION TABLE FOR JUMPER SEQUENCE
               ESCFLG = 2
               RADIX = 10 (DECIMAL)
               IODATA = 0
```

This routine is called after the parameterized escape sequence head (ESC & j) to set keyboard jumpers has been received by the terminal. The function table pointer (RNGTA) and associated variables are set by this routine to handle the setting of the keyboard option switches as specfied by the escape sequence.

Set Latching Keys Escape Sequence Processor

```
     STLKYS (ZSTLKY) - SET LATCHING KEYS ESCAPE SEQUENCE

       ENTRY:  DON'T CARE

       EXIT :  A,H,L DESTROYED
               RNGTA -> FUNCTION TABLE FOR LATCHING KEYS SEQUENCE
               ESCFLG = 2
               RADIX = 10 (DECIMAL)
               IODATE = 0
```

This routine is called after the parameterized escape sequence head (ESC & k) to set the latching keys has been received by the terminal. The function table pointer (RNGTA) and associated variables are set by this routine to handle the setting of the latching keys as specified by the escape sequence.

Alphabetic and Numeric Character Check Routines

```
     ALPCHK (ZALPCK) - CHECK FOR ALPHA TYPE CHARACTER

     NUMCHK (ZNUMCK) - CHECK FOR NUMERIC TYPE CHARACTER

       ENTRY:  A = CHARACTER TO BE CHECKED

       EXIT :  Z => CHARACTER TYPE IS CORRECT
               NZ => CHARACTER TYPE IS NOT CORRECT
```

These routines are used by the field checking processor to verify that a character belongs in either the alphabetic or numeric class of characters. Normally, alphabetic characters are defined as the letters a-z, A-Z, and the space character; and numeric characters as the digits 0-9, period (.), comma (,), plus (+), and minus (-).

HP 2645A DATA COMMUNICATIONS/MAIN CODE INTERFACE

INTRODUCTION

The Data Comm module provides the access from the terminal to an external host. This module contains the necessary drivers to transmit and receive data on the data comm interfaces.

The data comm code will occupy the 4K region from 20 to 24K (50000 to 57777 octal) in the ROM space. In many cases, the code will require only 2K of the space.

INTERFACE SPECIFICATIONS

Entry Vectors

The entry vectors to the data comm routines will begin at location 50010 (octal). The entry vectors will consist of "jumps" (JMP) to their associated subroutines. The vectors will be ordered as follows:

Location	Name	Function
050010	INITDC	Initialization Routine
050013	INI2DC	Initialization Continuator
050016	DCMON	Monitoring Routine
050021	DCCTL	Control Routine
050024	DCTST	Self-Test Routine
050027	GETDC	Character Input Routine
050032	PUTDC	Character Output Routine
050035	GETBIN	Binary Input Routine
050040	STBIN	Start Binary Output Routine
050043	ENDBIN	End Binary Output Routine
050046	DCINTR	Data Comm Interrupt Handler

Local Variables

Local variables for the data comm subroutines will be stored in the RAM region 177200-177377 (octal). The definition of this area is at the discretion of the particular data comm routines used.

Fast Access RAM

The data comm subroutines may use sixty-four bytes (64) in the fast access RAM region at locations 110600-110677 (octal).

Communications Protocol Constants

The constants used for a particular communications protocol occupy the area in the Data Comm ROM space beginning at 050000 (octal). The constants to be stored there are as follows:

Location	Usage
050002	Character to be used as block trigger (e.g., DC1)
050003	Record Separator Character (e.g., US)
050004	Block Terminator Character (e.g., RS)
050005	Data Comm Jumpers Mask (set bit to 1 to inhibit alteration of jumper setting by escape sequence)
050006	Data Comm Jumpers Mask for keyboard jumper R (Set high order bit only)

Returns

In general, a non-error return is signified by setting the C-flag to false, and an error return, by setting the C-flag to true. Whenever a data comm error is returned, the data comm error flag (ERRFLG [DCMERR]) must be set to one (1). This flag will be cleared by the main code after a status request is made by the computer.

FUNCTIONAL DESCRIPTION

General

The Data Comm module can be separated into a number of functional sections: initialization, interrupt process, monitor, input process, output process, and controls. Two Data Comm modules are available for the HP 2645A: Basic Data Comm and Multi-Point Data Comm.

BASIC DATA COMM

The Basic Data Comm module uses the data comm buffer for storing input characters only. The buffer is set up as a circular buffer with a load pointer (DCSPTR) and an unload pointer (DCBPTR).

Initialization

The initialization acquires the data comm buffer from the main code module and initializes the various buffer pointers. The keyboard option switches are checked to configure the data comm for normal or main channel protocol.

Interrupt Process

The interrupt process is called to retrieve input from the data comm interface PCA. Input is stored in the data comm buffer and the buffer load pointer is updated. If a buffer overflow occurs, the data comm error flag (ERRFLG [DCMERR]) is set and an all ones (377 octal) entry is made into the buffer. The error flag is also set if a parity or overrun error occurs on the data comm interface.

The interrupt process strips out any nulls (0B) or rubout (377B) from the input stream unless the Transparent Mode (DCFLGS [TRNMOD]) or Binary Mode (DCFLGS [BINMOD]) flags are set.

Monitor

The monitor routine checks the various data communications control signals for any changes to reflect a change in mode (receive/transmit). The monitor routine is also used for timing various operations (e.g., break).

nput Process

The input process gets characters from the data comm buffer and returns the character to the calling routine. If the data comm buffer is empty, the data comm PCA is checked for a character ready condition. When the hardware handshake option is selected, the data comm interface is inhibited from interrupting. So, no characters are read by the interrupt routine. Instead, the input routine must explicitly poll the PCA for input characters.

The input process responds to an ENQ from the input with an ACK unless the transparent or Binary Mode flags are set. The ENQ character is not returned to the calling routine.

Additional checks are made for line turn around characters if the terminal is configured for the main channel protocol option.

Output Process

The output process transmits characters out the data comm PCA. If the terminal is in Receive Mode, a fail return occurs and the character is not transmitted. Additional checks are made for line turn around characters if the terminal is operating with the main channel protocol option.

ontrols

A number of control routines are provided to set various terminal modes (e.g., binary or normal) and other data communications control functions (e.g., break, and modem disconnect).

MULTI-POINT DATA COMM

The Multi-Point Data Comm module uses the data comm buffer both for input and output. The particular protocol implemented operates in a Half Duplex Mode only (i.e., the terminal can either send or receive, but not simultaneously). Normally, the data comm is set for Receive Mode and cannot go into Transmit Mode unless the proper control sequence is received from the host computer. In this manner, the terminal is the slave of the computer.

The Multi-Point module drives both the Asynchronous and Synchronous Multi-Point PCA's. The module can receive and transmit data in either ASCII or EBCDIC, but all data between the Multi-Point module and the other terminal modules are passed as ASCII characters. Binary data is passed without any translation.

Initialization

The data comm and keyboard option switches are used to configure the operating conditions for the Multi-Point module. The size of the data comm buffer acquired from the Main Code module is determined from the data comm option switches.

Interrupt Process

The interrupt process is called to handle both input and output. Data is exchanged between the data comm buffer and the data comm interface PCA. The interrupt routine communicates with the input and output routines via the data comm context block. The context block identifies the mode of operation (receive/transmit) and the state of the data comm buffer.

The interrupt process recognizes when the terminal is being addressed by the computer and performs the necessary handshake protocol. The terminal is set either to Receive or Transmit Mode according to whether a select or poll sequence is received from the computer.

The interrupt process handles all aspects of the data transfer between the terminal and the computer. On input, the block check character is computed and compared against the received value. A positive acknowledgement is made if the values are equal. Otherwise, a request for retransmission of the last data block is made. On output, the block check character is computed and appended to the data. The data block is saved in the data comm buffer, for a possible retransmission, until a positive acknowledgment is received. Translation between ASCII and EBCDIC occurs in the interrupt routine.

Monitor

The monitor routine is used only to set the transmit light on the keyboard.

Input Process

The input process extracts characters from the input buffer and returns them to the calling routine.

Output Process

The output routine adds characters to the data comm buffer. If the terminal is in Receive Mode, an error return occurs and the character is not loaded into the data comm buffer. If the data comm buffer is full, the output routine waits until space becomes available before returning. That is, the output process does not return until the character has been placed into the data comm buffer or an error has occurred.

Controls

Control routines are available to perform functions similar to the control functions for the Basic Data Comm module. Additionally, there are some control functions which perform unique functions for the Multi-Point module (i.e., Program Attention (PA) and Program Function (PF) similar to the IBM 3270).

SUBROUTINE SPECIFICATIONS

The subroutine descriptions will specify the parameters to be included in the registers on entry and the register results on exit. Any registers not specified in the exit list retain their entry values. The processor flags (S, Z, P, and C) are generally altered by the subroutines. Certain routines use the processor flags to signify different return conditions. These conditions are specified in the subroutine descriptions.

The names of the subroutines to be used within the data comm code block are included in the following specifications. The names within parentheses are the names that will be used by the external code blocks to reference the associated subroutines.

Initialization Routine

```
     INITDC (ZINIDC) - INITIALIZE DATA COMM

       ENTRY:  DON'T CARE

       EXIT :  A DESTROYED
               B,C = NUMBER OF CONTINGUOUS BYTES NEEDED
                         FOR DATA COMM BUFFER
```

This subroutine is called whenever power is initially turned on or a complete terminal reset is being performed. Interrupts will be disabled by the calling routine when this subroutine is called.

Initialization of the data comm will be continued by calling the initialization continuator routine. The buffer returned then will always start at a 256-byte boundary.

Any errors on initialization should be indicated on return from the initialization continuator.

nitilization Continuator

```
     INI2DC (ZIN2DC) - CONTINUE DATA COMM INITIALIZATION

       ENTRY:  D,E = STARTING ADDRESS OF DATA COMM BUFFER

       EXIT :  A DESTROYED
               NC => INITIALIZATION SUCCESSFUL
               C => DATA COMM ERROR DETECTED
                 ERRFLG(DCMERR) = 1
                 H,L -> FIRST ERROR MESSAGE SECTION
```

This routine is called after a call to ZINIDC is made. Interrupts will be disabled when this routine is called.

The buffer starting address returned will always have the buffer start at a 256-byte boundary (i.e., E = 0).

On error returns, the main code of the terminal will display the message and "hang" the terminal until the user presses the RESET TERMINAL key.

Monitoring Routine

```
     DCMON (ZDCMON) - MONITORING ROUTINE

       ENTRY:  DON'T CARE

       EXIT :  ALL REGISTERS DESTROYED
```

This subroutine will be called approximately once every 10 milliseconds. The purpose of the subroutine is to provide for necessary periodic scanning of the data comm interface. If scanning of the data comm interface is not needed, then the subroutine should simply return without doing anything.

Control Routine

DCCTL (ZDCCTL) - DATA COMM CONTROL

```
  ENTRY:  A = CONTROL PARAMETER (see below)
          B-L, CONTROL VARIABLES (as required)

  EXIT :  A DESTROYED
          NC => NO DATA COMM ERRORS DETECTED
            Z => CONTROL PERFORMED SUCCESSFULLY
            NZ => INVALID CONTROL REQUEST
          C => DATA COMM ERROR DETECTED
            ERRFLG(DCMERR) = 1
            Z => NO ERROR MESSAGE
            NZ => DISPLAY ERROR MESSAGE
              H,L -> FIRST ERROR MESSAGE SECTION
```

This routine is used to perform various control functions on the data comm interface. Interrupts may be disabled for no more than 300 microseconds by any control routine. The control parameters and variables used are as follows:

Control
Parameter Function and Comments

0 CLRTRG - Clear Block Transfer Trigger Flag

Registers B-L are not used and should not be altered.

1 SETTRG - Set Block Transfer Trigger Flag

Registers B-L are not used and should not be altered.

2 RSETDC - Reset Data Comm

This call is used to cause the data comm routines to reset themselves to their initial condition. This control will be invoked when a soft terminal reset is performed. Registers B-L are not used and need not be saved.

3 SETREM - Set Remote Mode

This control is used when the user presses the REMOTE key to cause the terminal to go on-line. Registers B-L are not used and need not be saved.

4 SETLCL - Set Local Mode

This control is used when the user presses the REMOTE key to cause the terminal to go off-line. Registers B-L are not used and need not be saved.

5 PUTBRK - Output Break Signal

This control is used when the BREAK key is pressed by the user. If the break function is meaningless for the particular data comm protocol, a successful return will be given and the call ignored. Registers B-L are not used and need not be saved.

6 DISCNT - Modem Disconnect

This control is used to disconnect (hang up) the terminal from a dial-up connection by signalling the modem to turn off. If this function is meaningless for the particular data comm protocol, a successful return will be given and the call ignored.

7 ENDBLK - Terminal Output Message

This control is used to inform the data communications output routine that the last character of the current output message (block) has been sent. Note: An ENDBLK control is implied when a call is made to the terminate binary output routine (ENDBIN).

8 SETMON - Enter Monitor Mode

This control causes the data comm input and interrupt routine to pass all codes entered from the data comm. No handshaking on codes (e.g., ENQ/ACK) should be done by the data comm routines. As a result, some data comm protocols will disable the terminal from communicating with the CPU when in Monitor Mode (e.g., multipoint).

9 SETNRM - Enter Normal Mode

This control causes the data comm input and interrupt routine to resume normal operation. For example, resume ignoring nulls and rubouts, and resume ENQ/ACK handshake.

10 FSTBIN - Enter Fast Binary Transmit Mode

This control is called when a binary file is to be transmitted to the host computer. Its primary use is for the loading of binary code into the HP 21XX Series computers. Generally, this control is applicable only for the standard asynchronous data communications module. When the control is received and strap F on the keyboard interface is out (i.e., KBJMPR(FSTSND) = 1), the data comm interface should be switched to operate at 9600 baud. The board resumes normal operation when the ENDBIN (entry at 050043B) routine is invoked on the Data Comm module.

11 SNDATN - Transmit Attention Code

This control is called to cause a single character to be sent to the host computer. Its principal use is to emulate the "PAn" keys of the IBM 3270 terminal.

The B-register contains the character that is to be sent out. If this control is meaningless for a protocol, the routine should return with the flags set to indicate invalid control request (NC, NZ).

12 SNDFCT - Send Function Data

This control is used to emulate the "PFn" keys on the IBM 3270. The B-register contains the character to be used as the header for the data to be sent to the CPU.

If this control is meaningless for a protocol, the routine should return with the flags set to indicate an invalid control request (NC, NZ).

13 PROMPT - Send Prompt Code

This routine is called whenever a "DC2" prompt or its equivalent is to be sent to the CPU. The sending of the "DC2" is controlled by straps D, G, and H, and by the setting of the BLOCK MODE key. In the standard data comm protocol, a CR(LF) is added if the terminal is operating in Line Mode (strap D in). Refer to the "HP 2645A Programmer's Reference Manual" for the specific details on when a "DC2" is transmitted and its format.

If this function is meaningless to a protocol, an invalid control request return (NC, NZ) should be made by the control routine.

Self-Test Routine

```
DCTEST (ZDCTST) - PERFORM DATA COMM SELF-TEST

  ENTRY:  DON'T CARE

  EXIT :  A,D-L DESTROYED
          H,L -> FIRST MESSAGE SECTION
          NC => SELF-TEST SUCCESSFUL
          C => SELF-TEST FAILED
            ERRFLG(DCMERR) = 1
```

This subroutine is called to perform a self-test on the data communications interface in the terminal. After returning to the main code, the message pointed to by registers H and L is displayed. If the self-test failed, the terminal will "hang" until the user presses the RESET TERMINAL key.

haracter Input Routine

```
GETDC (ZGETDC) - GET ONE DATA COMM CHARACTER

  ENTRY:  DON'T CARE

  EXIT :  NC => NO ERRORS DETECTED
            Z => CHARACTER AVAILABLE
              A = CHARACTER FROM DATA COMM
            NZ => NO CHARACTER AVAILABLE
              A # 0, WAIT
                = 0, END OF INPUT BLOCK
          C => DATA COMM ERROR DETECTED
            A DESTROYED
            ERRFLG(DCMERR) = 1
            Z => NO ERROR MESSAGE
            NZ => DISPLAY ERROR MESSAGE
              H,L -> FIRST ERROR MESSAGE SECTION
```

This subroutine is called whenever a character is needed from the data comm. The routine will return only seven (7) bit characters. If the data comm protocol indicates that the data is binary, the GETDC routine will mask out the high order bit of the byte and return the remaining bits as a normal character. Interrupts may not be disabled by the subroutine.

When the "WAIT" condition is returned, the calling routine will continue normal operation and retry, at a later time, to get a data comm character.

When an error message is to be displayed for an error return, the main code of the terminal will display the message and "hang" the terminal until the user presses the RESET TERMINAL key. Messages should be stored in ascending order. If no error message is indicated, the terminal will abort the current operation and return to the preset level.

Character Output Routine

```
PUTDC (ZPUTDC) - OUTPUT CHARACTER TO DATA COMM

  ENTRY:  A = CHARACTER TO BE OUTPUT
          NC => NORMAL CHARACTER
          C => LAST CHARACTER IN BLOCK

  EXIT :  A DESTROYED
          NC => NO ERRORS DETECTED
            Z => CHARACTER ACCEPTED
            NZ => WAIT
          C => DATA COMM ERROR DETECTED
            ERRFLG(DCMERR) = 1
            Z => NO ERROR MESSAGE
            NZ => DISPLAY ERROR MESSAGE
              B,C -> FIRST ERROR MESSAGE SECTION
```

This subroutine is called whenever a character is to be sent out the data comm. The routine will be given seven (7) bit characters to output. The PUTDC routine is responsible for adding any necessary parity bits. Interrupts may be disabled by the subroutine for no more than 300 microseconds.

When the "WAIT" condition is returned, the calling routine will retry, at a later time, to send the same character until the character is accepted or a data comm error is detected. If an error condition is returned, data transmission will be terminated. When an error message is to be displayed, the main code of the terminal will display the message and "hang" the terminal until the user presses the RESET TERMINAL key. Otherwise, the terminal will return to the preset level.

Binary Input Routine

```
GETBIN (ZGTBIN) - GET ONE BINARY DATA COMM CHARACTER

  ENTRY:  DON'T CARE

  EXIT :  NC => NO ERRORS DETECTED
            Z => CHARACTER AVAILABLE
              A = CHARACTER FROM DATA COMM
            NZ => NO CHARACTER AVAILABLE
              A # 0, WAIT
              A = 0, END OF INPUT BLOCK
          C => DATA COMM ERROR DETECTED
            A DESTROYED
            ERRFLG(DCMERR) = 1
            Z => NO ERROR MESSAGE
            NZ => DISPLAY ERROR MESSAGE
              H,L -> FIRST ERROR MESSAGE SECTION
```

This subroutine is called whenever a binary character is needed from the data comm. The routine will return the entire eight bits received for each character. Interrupts may not be disabled by the subroutine.

If the previous character fetch from the data comm was via the GETDC routine, the initial call to GETBIN will perform any necessary handshake to prepare the communications line for binary data.

When the "WAIT" condition is returned, the calling routine will continue normal operation and retry, at a later time, to get a data comm character.

When an error message is to be displayed for an error return, the main code of the terminal will display the message and "hang" the terminal until the user presses the RESET TERMINAL key. Messages should be stored in ascending order. If no error message is indicated, the terminal will abort the current operation and return to the base level.

Start Binary Output

```
     STBIN (ZSTBIN) - START BINARY OUTPUT

       ENTRY:  ANY STATE

       EXIT :  A DESTROYED
               NC AND Z
```

This subroutine is called when succeeding output bytes are to be transmitted in the binary format.

End Binary Output

```
     ENDBIN (ZNDBIN) - END BINARY OUTPUT

       ENTRY:  ANY STATE

       EXIT :  A-E DESTROYED
               NC AND Z
```

This subroutine is called to terminate the Binary Transmit Mode and to restore normal character transmission. This subroutine will also execute the proper ENDBLK control for the binary output string.

Data Comm Interrupt Handler

```
     DCINTR (ZDCINT) - DATA COMM INTERRUPT HANDLER

       ENTRY:  ANY STATE

       EXIT :  ALL FLAGS AND REGISTERS RESTORED TO
                 STATE WHEN ROUTINE WAS ENTERED
```

This subroutine will be called whenever an interrupt is caused by the data comm interface. When the subroutine is called, the interrupt system will be disabled. The subroutine must re-enable interrupts within 300 microseconds.

HP 2645A ALTERNATE I/O CODE INTERFACE SPECIFICATION

INTRODUCTION

The alternate I/O code provides access to devices other than the normally available devices on the HP 2645A. The alternate I/O allows for custom devices to be connected to the terminal. The device driver must conform to the specifications below, but the functional aspects of the driver are defined and written by the user.

The alternate I/O code is allocated a 4K space within the ROM space. The 4K region is split into two 2K areas: 24K to 26K (60000-63777 octal) and 38K to 40K (114000-117777 octal). If only 2K of space is required, the 2K region from 24K to 26K must be used. Since all addresses in an action table used by the Main Code module must be less than 32K (100000 octal), the above allocation allows for the definition of an action table to be used by the Main Code module if the addresses lie within the 24K to 26K area of the alternate I/O code space (see Functional Description for Main Code module).

INTERFACE SPECIFICATIONS

Entry Vectors

The entry vectors to the alternate I/O code routines are to be located starting at 60002 (octal). The entry vectors will generally consist of "jumps" (JMP) to their associated subroutines:

Location	Name	Function
060002	INI1ALT	Initialization Routine
060005	INI2ALT	Initialization Continuator
060010	ALTINT	Interrupt Processor
060013	ALTMON	Monitoring Routine
060016	ALT2BF	Input Routine
060021	BF2ALT	Output Routine
060024	ALTCTL	Control Routine
060027	STAALT	Status Routine
060032	MSGALT	Device Name Message

Returns

In general, a non-error return is signified by setting the C-flag to false, an error return, by setting the C-flag to true. Whenever an error or user interrupt is returned, the I/O error flag (IOCERR) in the I/O variables space (177517 octal) must be set to the ASCII code for the letter "F" or "U" respectively. All "F" returns require an error message to be returned as described in the individual subroutine specifications. This flag will be cleared by the main code's I/O processor.

Local Variables

Local variables for the alternate I/O subroutines may be in the RAM region 177120-177147 (octal). The definition of these twenty-four bytes (24) is at the discretion of the particular alternate I/O routine used.

Fast Access RAM

The alternate I/O subroutines may use twenty-one bytes (21) in the fast access RAM region at locations 110553-110577 (octal).

I/O Buffers

Data is transferred between I/O drivers one record at a time. Each record is passed in one of the two I/O buffers. Each buffer is 256 bytes long and has a status, type, and length (described in detail below). The locations of the buffers and associated variables are given in the following table.

Location	Name	Description
176000	IOBUF1	Start of First I/O Buffer
176377	--	End of First I/O Buffer
177472	B1STAT	Status for First I/O Buffer
177471	B1TYPE	Type for First I/O Buffer
177470	B1LEN	Length for First I/O Buffer
176400	IOBUF2	Start of Second I/O Buffer
176777	--	End of Second I/O Buffer
177467	B2STAT	Status for Second I/O Buffer
177466	B2TYPE	Type for Second I/O Buffer
177465	B2LEN	Length for Second I/O Buffer

Data (IOBUFn)

One to 256 bytes of data are placed in the buffer beginning with address IOBUFn. There is no restriction on the data.

Status (BnSTAT)

The six low order bits (1-40 octal) are assigned to individual devices; bit 4 (20 octal) is assigned to the alternate I/O device. Bit 6 (100 octal) is reserved for use by Edit Mode routines. Bit 7 (200 octal) is reserved for use by the cartridge tape driver to hold a buffer while the cartridge tape interrupt routine empties the buffer.

A buffer is free if its status is zero. Otherwise, the buffer is owned by all devices whose bits are on (non-zero).

Type (BnTYPE)

There are three types of records: data records, end of file records, and end of data records.

A data record has type minus one (-1 = 377 octal).

An end of file record has type zero (0). The first data byte is reserved for the file number, which is filled in by the output routine. An end of file record may contain additional bytes, but these bytes will not be displayed, printed, or sent to the data comm.

An end of data record has type one (1). No data bytes are associated with an end of data record (i.e., the buffer length and contents are "don't cares").

Length (BnLEN)

The length gives the number of valid data bytes in the buffer. If the length is zero, all 256 bytes contain valid data (i.e., there are no "zero-length" records).

FUNCTIONAL DESCRIPTION

General

The Alternate I/O Code module is primarily driven by the Device I/O Code module of the HP 2645A. The alternate I/O device is accessed as device number five (5) in the I/O control escape sequence. The INSERT LINE and INSERT CHAR keys are used to specify the alternate I/O device as "from" and "to" devices respectively for device selection from the keyboard.

The Alternate I/O Code module consist of six functional sections: initialization, monitor, interrupt process, input, output, and control.

Initialization

The initialization routine acquires a private buffer, if required, from the Main Code module and initializes the operating state of the alternate I/O device.

Monitor

The monitor routine is called every 10 milliseconds when a processor timer interrupt occurs. This routine allows the device driver to perform any necessary monitoring of the device control lines.

Interrupt Process

This routine is called when an interrupt is detected from the alternate I/O device. To be recognized as such, the alternate I/O interface PCA must interrupt on the cartridge tape interrupt line on the bus (ATN2) and respond with poll bit 6 (100 octal) set when a poll request is made by the processor. (See description of 2645A microprocessor PCA: "Processor (8080A-2) Module - 13255-91093".)

Input

On input, a device driver must find a free buffer (status = 0) and set its bit in the corresponding buffer status byte (BnSTAT) to claim the buffer. The device driver should return the buffer with the device's bit in "BnSTAT" still set so that the buffer will not be considered empty before the I/O transfer routine sets the status byte to the destination device(s).

Output

On output, a device driver turns off its bit in the buffer status when it is finished recording the buffer. The driver must not make any other change in the buffer, buffer type, or length. (Exceptions - in the event of an error, a driver may free the buffer by setting the buffer status to zero. Also, the first data byte of an end of file record may be overwritten with the file number.)

Control

The control module is called to perform various control functions on the I/O devices. The function performed for a given control code is at the discretion of the particular alternate I/O device driver.

SUBROUTINE SPECIFICATIONS

The subroutine descriptions will specify the parameters to be included in the registers on entry and the register results on exit. Any registers not specified in the exit list retain their entry values. The processor flags (S Z, P, and C) are generally altered by the subroutines. Certain routines use the processor flags to signify different return conditions. These conditions are specified in the subroutine descriptions.

The names of the subroutines to be used within the Alternate I/O module are included in the following specifications. The names within parentheses are the names that will be used by the external code blocks to reference the associated subroutines.

Initialization Routine

```
     INIALT (ZINIAL) - INITIALIZE ALTERNATE I/O DEVICE

       ENTRY:  DON'T CARE

       EXIT :  A DESTROYED
               B,C = NUMBER OF CONTIGUOUS BYTES NEEDED
                     FOR ALTERNATE I/O BUFFER
               D-L   DON'T CARE
```

This subroutine is called whenever power is turned on or a complete terminal reset is performed. Interrupts will be disabled by the calling routine when this subroutine is called.

Initialization of the alternate I/O device will be continued by calling the initialization continuator routine. The buffer returned may not necessarily start on a 256-byte boundary. Any errors on initialization should be indicated on return from the initialization continuator.

Initialization Continuator

```
     IN2ALT (ZIN2AL) - CONTINUE ALTERNATE I/O INITIALIZATION

       ENTRY:  D,E = STARTING ADDRESS OF ALTERNATE I/O
                 BUFFER

       EXIT :  A DESTROYED
               NC => INITIALIZATION SUCCESSFUL
               B-L   DON'T CARE
               C => INITIALIZATION FAILED
                 H,L -> FIRST ERROR MESSAGE SECTION
               B-E   DON'T CARE
```

This routine is called after a call to ZINIAL is made. Interrupts are disabled when this routine is called. The buffer starts at the address given in registers D and E, and extends in ascending addresses for the number of bytes specified on return from INIALT.

On error returns, registers H and L contain the value to be stored in location "MSGPT1". The contents of the H and L registers will be stored in "MSGPT1" by the main code. All other message pointers (i.e., MSGPT2-MSGPT8) must be set by the routine. The main code will display the error message and "hang" the terminal until the user presses the RESET TERMINAL key.

Interrupt Processor

INTALT (ZINTAL) - ALTERNATE I/O INTERRUPT PROCESSOR

ENTRY: PSW, H AND L PUSHED ONTO TOP OF STACK

EXIT : ALL FLAGS AND REGISTERS RESTORED TO
STATE WHEN INTERRUPT OCCURRED

This subroutine is called whenever an interrupt is caused by the alternate I/O device. When the subroutine is called, the interrupt will be disabled. The subroutine must re-enable interupts within 300 microseconds.

Monitoring Routine

MONALT (ZMONAL) - MONITOR ALTERNATE I/O DEVICE

ENTRY: DON'T CARE

EXIT : ALL REGISTERS DESTROYED

This subroutine is called approximately once every 10 milliseconds. The purpose of the subroutine is to provide for necessary periodic scanning of the alternate I/O interface. If scanning of the interface is not needed, then the subroutine should simply return without doing anything.

Control Routine

CTLALT (ZCTLAL) - ALTERNATE I/O CONTROL

```
  ENTRY:  IOCTYP = CONTROL TYPE CODE NUMBER
          IOCCNT = CONTROL PARAMETER (2 BYTES)
          IOPSGN = SIGN OF CONTROL PARAMETER
                 = +1 => POSITIVE
                 = 200 => NO SIGN
                 = -1 => NEGATIVE

  EXIT :  NC => CONTROL PERFORMED SUCCESSFULLY
            IOCERR = "S" => SUCCESSFUL COMPLETION
          C => CONTROL FUNCTION ABORTED
            IOCERR = "F" => ABORTED ON FAILURE
              MSGPT1-MSGPT8 = ERROR MESSAGE STRING(S)
            IOCERR = "U" => USER INTERRUPTED OPERATION
          ALL REGISTERS DESTROYED
```

This routine is used to perform various control functions on the alternate I/O device. Interrupts may be disabled for no more than 300 microseconds by any control routine. The actual function performed for a given control code is defined by the specific alternate I/O driver installed in the terminal. The functions performed for the Cartridge Tape Units (CTU's) are as follows:

Control Code	Function
0	Rewind
1	Space "p" records
2	Space "p" files
3	Locate end-of-data mark
4	Condition tape
5	Record file mark
6	Record end-of-data mark
7	Test cartridge tape unit
8	Skip "p" records immediately without recording end-of-data mark
9	Enter Write-Backspace-Read Mode
10	Exit Write-Backspace-Read Mode

where "p" is the combined value of IOPSGN and IOCCNT

Functions implemented for the alternate I/O device should attempt to perform similar functions for corresponding control codes.

Status Routine

```
     STAALT (ZSTAAL) - GET ALTERNATE I/O STATUS

       ENTRY:  DON'T CARE

       EXIT :  NC
               IOSTA1,IOSTA2,IOSTA3 = CURRENT DEVICE
                 STATUS
               ALL REGISTERS DESTROYED
```

This routine is called to extract the current device status to be sent to the host computer. The status is returned in the lower four (4) bits of each status byte (IOSTA1-IOSTA3). The definition of each bit is left to the specific alternate I/O device. An attempt should be made to use the same status bit as used in existing devices for common status conditions (e.g., read error status).

Input Routine

```
     ALT2BF (ZGETAL) - ALTERNATE I/O INPUT ROUTINE

       ENTRY:  D,E -> STATUS OF LAST BUFFER RETURNED
                 (DON'T CARE FOR FIRST READ)

       EXIT :  A,B,C,H,L DESTROYED
               NC => SUCCESSFUL READ
                 D,E -> BUFFER STATUS
               C => ERROR
                 IOCERR = U => USER INTERRUPTED
                 IOCERR = F => FAILURE
                   MSGPTX -> ERROR MESSAGE
                 D,E DESTROYED
```

It is the responsibility of the input routine to find a free buffer (status = 0). For a successful return, the alternate I/O bit (bit 4 -20 octal) should be turned on in the buffer status (BNSTAT). The buffer type and length must also be correctly set (see "I/O Buffers", above).

The input routine is also responsible for checking for user interrupt (RETURN key) by calling ZGETKY (see keyboard code interface). To avoid loss of data, this check should be done before each record is read from the alternate I/O device. Note that no message is returned for user interrupts.

An error message must be supplied for all failure returns. It is not necessary to free buffers claimed before the error is detected as all buffers are freed by the main code on any error returns.

Output Routine

```
     BF2ALT (ZPUTAL) - ALTERNATE I/O OUTPUT ROUTINE

       ENTRY:  D,E -> BUFFER STATUS
               ALTERNATE I/O BIT SET IN BUFFER STATUS

       EXIT :  A,B,C,H,L DESTROYED
               NC => SUCCESS
                 D,E -> BUFFER STATUS
               C => FAILURE
                 IOCERR = F
                 MSGPTX -> ERROR MESSAGE
               D,E DESTROYED
```

For a successful return, this routine must not alter the buffer, buffer type, or buffer length at any time. When the alternate I/O driver is finished with the buffer, the driver should release the buffer by turning off the alternate I/O bit (bit 4 - 20 octal) in the buffer status.

BF2ALT may alter the buffer and its associated variables in any way in the event of an error. (All buffers are freed by the main code on any error return.) A message must be supplied with every error return.

To avoid loss of data, BF2ALT should not abort on user interrupts (RETURN key).

Device Name Message

```
     MSGALT (ZMSGAL) - DEVICE NAME
```

A string representing the name of the alternate I/O device is stored starting at location 60032 (octal). The string is stored in order of ascending addresses The device name message is used by the compare routine to report a data terminator (end of file or end of data) on the alternate I/O device during a compare operation. The message is optional if ALT2BF can only return data records and not an end of file or end of data or if the alternate I/O device is an output only device. The message should have the following format:

```
    MSGALT  DEF  ' ON <device name>',0
```

Note the preceding blank and trailing null are required as part of the message.

COMMON AREA ALLOCATION

The common area is located in the region 177720-177777 (octal). The variables stored there are as follows:

Location	Name	Usage
177776	DISPST	Display Refresh Starting Pointer (2 bytes)
177775	TRMTYP	Terminal Type Number
177774	KBDCSW	Keyboard Data Comm Switch Settings
177773	KBJMPR	Keyboard Jumpers A-B-C-D-E-F-G-H
177772	KBJMP2	Keyboard Jumpers J-K-L-M-N-P-Q-R
177771	KBJMP3	Keyboard Jumpers S-T-U-V-W-X-Y-Z
177770	CMFLGS	Common Flags
177767	ERRFLG	Error Flags
177766	INTFLG	Interrupt Flag
177765	PRCCIL	Processor Control Flags
177764	MDFLG1	Terminal Mode Flags 1
177763	MDFLG2	Terminal Mode Flags 2
177761	MSGPT1	First Message Block Pointer (2 bytes)
177757	MSGPT2	Second Message Block Pointer (2 bytes)
177755	MSGPT3	Third Message Block Pointer (2 bytes)
177753	MSGPT4	Fourth Message Block Pointer (2 bytes)
177751	MSGPT5	Fifth Message Block Pointer (2 bytes)
177747	MSGPT6	Sixth Message Block Pointer (2 bytes)
177745	MSGPT7	Seventh Message Block Pointer (2 bytes)
177743	MSGPT8	Eighth Message Block Pointer (2 bytes)
177741	CTIVEC	CTU Interrupt Vector (2 bytes)
177740	CTIJMP	"JMP" Instruction for Vector
177736	IODATA	ESC Sequence Parameter Accumulator (2 bytes)
177735	IOCSGN	Sign for Parameter
177734	IOPSGN	Sign for Assigned Parameter
177733	PARM1	Assigned Parameter 1
177732	PARM2	Assigned Parameter 2
177731	PARM3	Assigned Parameter 3
177730	PARM4	Assigned Parameter 4
177727	PARM5	Assigned Parameter 5
177725	PARM6	Assigned Parameter 6 (2 bytes)
177724	RADIX	Radix for Accumulating Parameters
177722	RNGTA	Character Function Table Address (2 bytes)
177721	ESCFLG	Escape Sequence in Progress Flag
177720	RSTTMR	Soft Reset Timer

DEFINITION

DISPST Contains the two byte link address pointing to the top line to be displayed.

TRMTYP Contains the terminal type code number: 0 <= TRMTYP < 16

Bit	Meaning
0	I/O Firmware Installed 0 = Not installed 1 = Installed
1	Bi-lingual Terminal 0 = Terminal is Roman type only 1 = Terminal has bi-lingual functions
2	Reserved
3	Unused
4-7	Always zero (0)

KBDCSW Contains the settings of the Duplex, Parity, and Baud Rate switches on the keyboard.

Bit	Meaning
0	Undefined
1-3	Baud Rate 0 = Ext 1 = 110 2 = 150 3 = 300 4 = 1200 5 = 2400 6 = 4800 7 = 9600
4-5	Parity 0 = Even 1 = Odd 2 = None 3 = Undefined
6	Undefined
7	Duplex 0 = Half 1 = Full

KBJMPR, KBJMP2, KBJMP3

Contains the settings of the jumpers on the keyboard interface PCA. In all cases, the bit is set to one when the jumper is out, and zero, when the jumper is in.

```
    Bit  Meaning

KBJMPR
     0   CONDIS - Transmit All Function Keys
           0 = Disabled
           1 = Enabled
     1   SPLDIS - Space Overwrite Latch
           0 = Disabled
           1 = Enabled
     2   LINwRP - Cursor End-of-Line Wrap Around
           0 = Enabled
           1 = Disabled
     3   PAGSTR - Line/Page Mode
           0 = Line
           1 = Page
     4   LFPOS - Location of Line Feed for Remote READ
           0 = Line Feed at beginning of record
           1 = Line Feed at end of record
     5   FSTSND - Fast Binary Read
           0 = Disabled
           1 = Enabled
     6   HNDSHK - Handshake
           0 = Disabled
           1 = Enabled
     7   DC2SND - Inhibit DC2
           0 = Disabled
           1 = Enabled

KBJMP2
     0   AUTEND - Add Terminator on "ENTER"
           0 = Disabled
           1 = Enabled
     1   CLRTRM - Clear Terminator After Display Sent
           0 = Disabled
           1 = Enabled
     2   NOTEST - Inhibit Terminal Self-Test
           0 = Self-Test enabled
           1 = Self-Test disabled
     3   EDTwRP - Invert Edit wrap Around Control
           0 = Disabled
           1 = Enabled
     4   PRNTAL - Send All Enhancement Codes to Printer
           0 = Disabled
           1 = Enabled
    5-6  Undefined
     7   DCJMP0 - Reserved for Data Comm Usage
```

```
KBJMP3
       0    DCJMP1 - Reserved for Data Comm Usage
       1    DCJMP2 - Reserved for Data Comm Usage
       2    DCJMP3 - Reserved for Data Comm Usage
       3    DCJMP4 - Reserved for Data Comm Usage
       4    NUDCST - Inhibit Data Comm Self-Test
              0 = Data comm self-test enabled
              1 = Data comm self-test disabled
       5    SETCH - Turn on "CH" Control Line
              0 = CH off
              1 = CH on
       6    CHEKCC - Monitor "CC" Control Line
              0 = Monitor normal transmit indicator
              1 = Set transmit indicator according to CC
       7    FRCPTY - Force Parity/No Input Parity Check
              0 = Use normal parity
              1 = Enable special parity
```

```
CMFLGS  Individual bits represent various modes of the terminal.

       Bit  Meaning

       0    BLKTRG - Block Transfer Trigger
              0 = Clear
              1 = Set
       1    INSWRP - Insert with Wrap Around
              0 = Wrap around disabled
              1 = Wrap around enabled
       2    FRCRST - Force Full Terminal Reset if Reset
              0 = Do soft reset only
              1 = Perform full reset
       3    DEFSKY - Define Soft Key Mode
              0 = Normal Terminal Mode enabled
              1 = Soft key menu enabled
       4    REMSET - Remote/Local Mode
              0 = Terminal in Local Mode
              1 = Terminal in Remote Mode
       5    RCVMDE - Data Comm Mode
              0 = Transmit
              1 = Receive
       6    ETXRCV - End of Input
              0 = End of text input flag not received
              1 = End of text input flag received
       7    UNUSED
```

ERRFLG Each bit represents an error condition to be displayed as part of the terminal status.

Bit Meaning

0 DCMERR - Data Comm Error
0 = No Data Comm Errors
1 = Data Comm Error(s) Detected

1 TESTOK - Terminal Self-Test
0 = Malfunction Detected
1 = No Malfunctions

2 LDRCHK - Loader Checksum
0 = Checksum Error in Loading Sequence
1 = No Error

3-7 Undefined

INTFLG Indicates if the timer caused an interrupt. INTFLG is set to 3 for a timer interrupt. All other interrupts do not change the value of INTFLG.

PRCCTL Contains the current state of the processor board.

Bit Meaning

0 Undefined

1 TMRON - 10 Millisecond Timer On
0 = Timer off
1 = Timer on

2 TMIEN - Timer Interrupt Acknowledge/Reset
0 = Acknowledge timer interrupt
1 = Re-enable timer interrupt

3 DCIOFF - Data Comm Interrupt
0 = Enabled
1 = Disabled

4 TIMOFF - Timer Interrupt
0 = Enabled
1 = Disabled

5 POLL - ATN2 Poll
0 = Disabled
1 = Enabled

6 Undefined

7 SETROM - ROM/RAM Enabled
0 = ROM enabled, RAM disabled
1 = RAM enabled, ROM disabled

MDFLG1 Contains the first set of Terminal Mode flags. The flags generally refer to modes that are electronically latched.

Bit	Meaning (0=>DISABLED, 1=>ENABLED)
0	DSPFNC - Display Functions Enabled
1	INSCHR - Insert Character Enabled
2	MEMLOK - Memory Lock Enabled
3	FORMAT - Format Mode Enabled
4	EDIT - Edit Mode Enabled
5	SELECT - Device Select Mode
6	RECORD - Record Mode Enabled
7	FORGN - Foreign Mode Enabled

MDFLG2 Contains the second set of Terminal Mode flags. The flags generally refer to modes that are set by latching keys.

Bit	Meaining
0	CAPSLK - Caps Lock Enabled
1	BLKMDE - Block Mode Enabled
2	AUTOLF - Automatic Line Feed Enabled
3	REMOTE - Remote Enabled
4	WBSR - WRITE-BACKSPACE-READ MODE ENABLED
5-7	UNUSED

MSGPT1, MSGPT2, MSGPT3, MSGPT4, MSGPT5, MSGPT6, MSGPT7, MSGPT8

These two byte values are used to store pointers to message blocks for the message display routine (DSPMSG).

CTIVEC Contains the start address for the current cartridge tape interrupt routine.

CTIJMP Contains the operation code for the "JMP" instruction (303 octal).

IODATA Accumulator for parameters specified in parameterized escape sequences. As the numbers for the parameters are received, the accumulated value is maintained in this two-byte location. The base of the value is specified in location "RADIX".

IOCSGN Contains the sign of the parameter currently being received for a parameterized escape sequence.

Value	Meaning
+1	Sign is positive
0	No value specified for parameter
200B	No sign specified for parameter value
-1	Sign is negative

IOPSGN Contains the sign of the parameter that has been received. Before a parameter is to be evaluated by the "CHKLIM" routine, the sign of the parameter must be placed in this location. The values assigned are the same as "IOCSGN".

PARM1, PARM2, PARM3, PARM4, PARM5, PARM6

These locations are used as work areas to store the parameters for the various parameterized escape sequences. The usage for each location is defined by the particular escape sequence handler.

RADIX This value is set to the radix of the numbers to be entered as values for a parameterized escaped sequence.

RNGTA Contains the pointer to the currently active "action table" defining the function of characters entered into the terminal.

ESCFLG This location is set to all 1's if an escape sequence is currently being processed from the data comm while operating in Block Mode. Otherwise, the value is zero. When the value is all 1's, the keyboard is locked out.

RSTTMR When a soft reset is executed, this location is set to the number of 10 millisecond intervals during which a full reset will occur if the RESET button is pressed. If the value is zero, a soft reset will occur (unless CMFLGS(FRCRST) is set to 1).

```
=========================================================================
ITEM   LOC   OBJECT CODE  SOURCE STATEMENTS                        PAGE  1
=========================================================================
   1   0000  .    .    .            ASB,HEX   ;PT774 - 11/10/76 - 1130 HOURS
   2   0000  .    .    .  ;**********************
   3   0000  .    .    .  ; VERSION LEVEL CODE *
   4   0000  .    .    .  ;**********************
   5   0050  .    .    .  VERSN  EQU  120Q      ;P => VERSION 0
   6   0051  .    .    .  VERSN1 EQU  121Q      ;Q => VERSION 1
   7   0000  .    .    .  ;
   8   0000  .    .    .  ;  NOTE:  THE SECOND ROM WAS RE-ORDERED TO FIX
   9   0000  .    .    .  ;  A BUG, SO ONLY THAT ROM HAS VERSION NUMBER 1.
  10   0000  .    .    .  ;
  11   0000  .    .    .  ;
  12   0000  .    .    .  ;  COMMON EQUATES - CM35 - 6/27/76 - 1830 HOURS
  13   0000  .    .    .  ;
  14   9100  .    .    .  FSTRAM EQU  110400Q   ;FAST RAM LOWER LIMIT
  15   0000  .    .    .  ;****************************************
  16   0000  .    .    .  ; KBDCSW - KEYBOARD DATA COMM SWITCHES *
  17   0000  .    .    .  ;****************************************
  18   0080  .    .    .  FULDUP EQU  200Q      ;HALF/FULL DUPLEX
  19   0000  .    .    .  ;***************************************
  20   0000  .    .    .  ; KBJMPR - KEYBOARD INTERFACE JUMPERS *
  21   0000  .    .    .  ;***************************************
  22   0000  .    .    .  ;
  23   0000  .    .    .  ;  JUMPERS SENSED AS 0' WHEN INSERTED
  24   0000  .    .    .  ;
  25   0000  .    .    .  ;  ALL JUMPERS ARE NORMALLY INSERTED
  26   0000  .    .    .  ;
  27   0001  .    .    .  CONDIS EQU 001Q     ;CONTROL CODE DISABLE
  28   0000  .    .    .  ;                     (0=DISABLED)
  29   0002  .    .    .  SPLDIS EQU 002Q     ;SPOW LATCH DISABLE
  30   0000  .    .    .  ;                     (0=DISABLED)
  31   0004  .    .    .  LINWRP EQU 004Q     ;COLUMN 80 AUTO CR,LF
  32   0000  .    .    .  ;                     (0=ENABLED)
  33   0008  .    .    .  PAGSTR EQU 010Q     ;PAGE MODE STRAP
  34   0000  .    .    .  ;                     (0=LINE-FIELD MODE)
  35   0010  .    .    .  LFPOS  EQU 20Q      ;LINE FEED POSITION
  36   0000  .    .    .  ;                     (0 = POSITION LINE FEED
  37   0000  .    .    .  ;                          AT START OF NEXT I/O
  38   0000  .    .    .  ;                          READ
  39   0000  .    .    .  ;                      1 = PUT LINE FEED AT END
  40   0000  .    .    .  ;                          OF RECORD)
  41   0020  .    .    .  FSTSND EQU 40Q      ;9600 BAUD DATACOM SHIFT
  42   0000  .    .    .  ;                     (0=9600 BAUD FOR ESC,E)
  43   0040  .    .    .  HNDSHK EQU 100Q     ;BLOCK TRANSFER HANDSHAKE
  44   0000  .    .    .  ;                     (0 = FOLLOW DC2SND SETTING
  45   0000  .    .    .  ;                      1 = SEND DC2 BEFORE DATA)
  46   0080  .    .    .  DC2SND EQU 200Q
  47   0000  .    .    .  ;                     (0 = SEND DC2 ON ENTER
  48   0000  .    .    .  ;                           AND FUNCTION KEY IN
  49   0000  .    .    .  ;                           BLOCK MODE
  50   0000  .    .    .  ;                      1 = INHIBIT ALL DC2
  51   0000  .    .    .  ;                           HANDSHAKE)
```

```
===========================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                       PAGE   2
===========================================================================
   53   0000    .    .    .  ;*******************************************
   54   0000    .    .    .  ; KBJMP2 - SECOND SET OF KEYBOARD JUMPERS *
   55   0000    .    .    .  ;*******************************************
   56   0001    .    .    .  AUTTRM EQU  1Q         ;AUTO TERMINATE ON "ENTER"
   57   0002    .    .    .  CLRTRM EQU  2Q         ;CLEAR TERMINATOR ON TRANSMI
   58   0004    .    .    .  NOTEST EQU  4Q         ;INHIBIT TERMINAL SELF-TEST
   59   0008    .    .    .  EDTWRP EQU  10Q        ;INVERT SENSE OF EDIT WRAP
   60   0010    .    .    .  PRNTAL EQU  20Q        ;SEND ALL CODES TO PRINTER
   61   0080    .    .    .  DCJMP0 EQU  200Q       ;DATA COMM JUMPER
   62   0000    .    .    .  ;******************************************
   63   0000    .    .    .  ; KBJMP3 - THIRD SET OF KEYBOARD JUMPERS *
   64   0000    .    .    .  ;******************************************
   65   0001    .    .    .  DCJMP1 EQU  1Q         ;DATA COMM JUMPERS
   66   0002    .    .    .  DCJMP2 EQU  2Q               ;.
   67   0004    .    .    .  DCJMP3 EQU  4Q               ;.
   68   0008    .    .    .  DCJMP4 EQU  10Q              ;.
   69   0010    .    .    .  NODCST EQU  20Q        ;INHIBIT DATA COMM SELF-TEST
   70   0000    .    .    .  ;                        (0 = DISABLED)
   71   0020    .    .    .  SETCH  EQU  40Q        ;TURN ON "CH" CONTROL LINE
   72   0000    .    .    .  ;                        (0 = OFF, 1 = ON)
   73   0040    .    .    .  CHEKCC EQU  100Q       ;MONITOR CC CONTROL LINE
   74   0000    .    .    .  ;                        (1 = ENABLED)
   75   0080    .    .    .  FRCPTY EQU  200Q       ;FORCE PARITY ON/NO IN CHECK
   76   0000    .    .    .  ;                        (1 = ENABLED)
   77   0000    .    .    .  ;************************
   78   0000    .    .    .  ; CMFLGS - COMMON FLAGS *
   79   0000    .    .    .  ;************************
   80   0001    .    .    .  BLKTRG EQU  1Q         ;BLOCK TRANSFER TRIGGER
   81   0002    .    .    .  INSWRP EQU  2Q         ;INSERT WITH WRAP AROUND
   82   0004    .    .    .  FRCRST EQU  4Q         ;FORCE FULL TERMINAL RESET
   83   0008    .    .    .  DEFSKY EQU  10Q        ;DEFINE SOFT KEY MODE ENABLE
   84   0010    .    .    .  REMSET EQU  20Q        ;REMOTE MODE ENABLED
   85   0020    .    .    .  RCVMDE EQU  40Q        ;TERMINAL IN RECEIVE MODE
   86   0000    .    .    .  ;***********************
   87   0000    .    .    .  ; ERRFLG - ERROR FLAGS *
   88   0000    .    .    .  ;***********************
   89   0001    .    .    .  DCMERR EQU  1Q         ;DATACOM (1 = ERROR)
   90   0002    .    .    .  TESTOK EQU  2Q         ;SELF-TEST (0 = ERROR)
   91   0004    .    .    .  LDRCHK EQU  4Q         ;LOADER CHECKSUM (0 = ERROR)
   92   0000    .    .    .  ;**************************
   93   0000    .    .    .  ; INTFLG - INTERRUPT FLAG *
   94   0000    .    .    .  ;**************************
   95   0003    .    .    .  TMRINT EQU  3          ;TIMER INTERRUPT
```

```
================================================================================
 ITEM    LOC    OBJECT CODE   SOURCE STATEMENTS                             PAGE   3
================================================================================
   97    0000    .    .    .   ;*************************************
   98    0000    .    .    .   ; PRCCTL - PROCESSOR CONTROL FLAGS *
   99    0000    .    .    .   ;*************************************
  100    0000    .    .    .   TMIACK EQU  0Q          ;ACKNOWLEDGE TIMER INTERRUPT
  101    0000    .    .    .   ;                         (BIT 1 OFF)
  102    0001    .    .    .   TMRON  EQU  1Q          ;SET TIMER ON
  103    0002    .    .    .   TMIEN  EQU  2Q          ;RE-ENABLE TIMER INTERRUPT
  104    0010    .    .    .   DCIOFF EQU  20Q         ;DISABLE DATA COMM INTERRUPT
  105    0020    .    .    .   TMIOFF EQU  40Q         ;DISABLE TIMER INTERRUPTS
  106    0040    .    .    .   POLL   EQU  100Q        ;POLL CTU INTERRUPTS
  107    0080    .    .    .   SETROM EQU  200Q        ;DISABLE (1)/ENABLE (0) ROM
  108    0000    .    .    .   ;**********************************
  109    0000    .    .    .   ; MDFLG1 - TERMINAL MODE FLAGS 1 *
  110    0000    .    .    .   ;**********************************
  111    0001    .    .    .   DSPFNC EQU  1Q          ;DISPLAY FUNCTIONS ENABLED
  112    0002    .    .    .   INSCHR EQU  2Q          ;INSERT CHARACTER ENABLED
  113    0004    .    .    .   MEMLOK EQU  4Q          ;MEMORY LOCK ENABLED
  114    0008    .    .    .   FORMAT EQU  10Q         ;FORMAT MODE ENABLED
  115    0010    .    .    .   EDIT   EQU  20Q         ;EDIT MODE ENABLED
  116    0020    .    .    .   SELECT EQU  40Q         ;SELECT MODE ENABLED
  117    0040    .    .    .   RECORD EQU  100Q        ;RECORD MODE ENABLED
  118    0080    .    .    .   FORGN  EQU  200Q        ;FOREIGN MODE ENABLED
  119    0000    .    .    .   ;**********************************
  120    0000    .    .    .   ; MDFLG2 - TERMINAL MODE FLAGS 2 *
  121    0000    .    .    .   ;**********************************
  122    0001    .    .    .   CAPSLK EQU  1Q          ;CAPS LOCK ENABLED
  123    0002    .    .    .   BLKMDE EQU  2Q          ;BLOCK MODE ENABLED
  124    0004    .    .    .   AUTOLF EQU  4Q          ;AUTO LF ENABLED
  125    0008    .    .    .   REMOTE EQU  10Q         ;REMOTE ENABLED
  126    0020    .    .    .   WBSR   EQU  40Q         ;WRITE-BACKSPACE-READ MODE
  127    0000    .    .    .   ;**********************************************
  128    0000    .    .    .   ; RADIX - BASE OF INPUT PARAMETER FOR ESC SEQ *
  129    0000    .    .    .   ;**********************************************
  130    000A    .    .    .   DECRDX EQU  10          ;DECIMAL NUMBERS
  131    0008    .    .    .   OCTRDX EQU  8           ;OCTAL NUMBERS
```

```
0000   .   .   .    ;*******************
0000   .   .   .    ; COMMON VARIABLES *
0000   .   .   .    ;*******************
9165   .   .   .    INTVEC EQU  FSTRAM+145Q  ;CENTRAL INTERRUPT VECTOR
9168   .   .   .    SCNVEC EQU  INTVEC+3  ;FOREIGN TERMINAL DISPLY SCA
0000   .   .   .    ;
FFFF   .   .   .    COMMON EQU  177777Q   ;UPPER LIMIT OF COMMON AREA
00FF   .   .   .    CMBASE EQU  COMMON/256  ;MSB OF COMMON ADDRESSES
FF00   .   .   .    CMSTOR EQU  CMBASE*256  ;MSB ADJUSTMENT FACTOR
0000   .   .   .    ;
FFFE   .   .   .    DISPST EQU  COMMON-1  ;DISPLAY REFRESH START PTR
FFFD   .   .   .    TRMTYP EQU  DISPST-1  ;TERMINAL TYPE NUMBER
FFFC   .   .   .    KBDCSW EQU  TRMTYP-1  ;KEYBOARD DATACOM SWITCHES
FFFB   .   .   .    KBJMPR EQU  KBDCSW-1  ;KEYBOARD STRAPS
FFFA   .   .   .    KBJMP2 EQU  KBJMPR-1    ;SET 2
FFF9   .   .   .    KBJMP3 EQU  KBJMP2-1    ;SET 3
FFF8   .   .   .    CMFLGS EQU  KBJMP3-1  ;COMMON FLAGS
FFF7   .   .   .    ERRFLG EQU  CMFLGS-1  ;ERROR FLAGS
FFF6   .   .   .    INIFLG EQU  ERRFLG-1  ;INTERRUPT FLAG
FFF5   .   .   .    PRCCTL EQU  INIFLG-1  ;PROCESSOR CONTROL FLAGS
FFF4   .   .   .    MDFLG1 EQU  PRCCTL-1  ;TERMINAL MODE FLAGS 1
FFF3   .   .   .    MDFLG2 EQU  MDFLG1-1    ;AND 2
FFF1   .   .   .    MSGPT1 EQU  MDFLG2-2  ;MESSAGE POINTERS
FFEF   .   .   .    MSGPT2 EQU  MSGPT1-2         ;.
FFED   .   .   .    MSGPT3 EQU  MSGPT2-2         ;.
FFEB   .   .   .    MSGPT4 EQU  MSGPT3-2         ;.
FFE9   .   .   .    MSGPT5 EQU  MSGPT4-2         ;.
FFE7   .   .   .    MSGPT6 EQU  MSGPT5-2         ;.
FFE5   .   .   .    MSGPT7 EQU  MSGPT6-2         ;.
FFE3   .   .   .    MSGPT8 EQU  MSGPT7-2         ;.
FFE1   .   .   .    CTIVEC EQU  MSGPT8-2  ;CTU INTERRUPT VECTOR
FFE0   .   .   .    CTIJMP EQU  CTIVEC-1  ;JUMP CODE FOR VECTOR
FFDE   .   .   .    IODATA EQU  CTIJMP-2  ;ESQ SEQ PARM ACCUMULATOR
FFDD   .   .   .    IOCSGN EQU  IODATA-1  ;SIGN FOR PARAMETER
FFDC   .   .   .    IOPSGN EQU  IOCSGN-1  ;PARAMETER SIGN
FFDB   .   .   .    PARM1  EQU  IOPSGN-1  ;ESCAPE SEQUENCE PARAMETERS
FFDA   .   .   .    PARM2  EQU  PARM1-1          ;.
FFD9   .   .   .    PARM3  EQU  PARM2-1          ;.
FFD8   .   .   .    PARM4  EQU  PARM3-1          ;.
FFD7   .   .   .    PARM5  EQU  PARM4-1          ;.
FFD5   .   .   .    PARM6  EQU  PARM5-2          ;.
FFD4   .   .   .    RADIX  EQU  PARM6-1   ;RADIX OF PARAMETERS
FFD2   .   .   .    RNGTA  EQU  RADIX-2   ;CHAR FUNCTION TABLE ADDRESS
FFD1   .   .   .    ESCFLG EQU  RNGTA-1   ;ESCAPE SEQUENCE FLAG
0000   .   .   .    ;                       = 0, NOT IN ESCAPE SEQ
0000   .   .   .    ;                       # 0, ESC SEQ IN PROGRESS
FFD0   .   .   .    RSTTMR EQU  ESCFLG-1  ;SOFT RESET TIMER
0000   .   .   .    ; * * * * * * * * * * * * * * * * * * * * * * * * *
0000   .   .   .    ;  END OF COMMON EQUATES                          *
0000   .   .   .    ;*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*^*
```

```
==========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                    PAGE   5
==========================================================================
  184   0000   .   .   .   ;*********************************
  185   0000   .   .   .   ; KEYBOARD ENTRY VECTOR POINTERS *
  186   0000   .   .   .   ;*********************************
  187   4800   .   .   .   ZKBBAS EQU  44000Q    ;KEYBOARD START ADDRESS
  188   4802   .   .   .   ZINIKB EQU  ZKBBAS+2  ;INITIALIZE KEYBOARD
  189   4805   .   .   .   ZGETKY EQU  ZINIKB+3  ;GET KEYBOARD KEY
  190   4808   .   .   .   ZKBCTL EQU  ZGETKY+3  ;PERFORM KEYBOARD CONTROL
  191   480B   .   .   .   ZKBMON EQU  ZKBCTL+3  ;MONITOR KEYBOARD
  192   480E   .   .   .   ZSTMD1 EQU  ZKBMON+3  ;SET MODE 1 FLAGS
  193   4811   .   .   .   ZCLMD1 EQU  ZSTMD1+3  ;CLEAR MODE 1 FLAGS
  194   4814   .   .   .   ZBELL  EQU  ZCLMD1+3  ;SOUND THE BELL
  195   4817   .   .   .   ZSTXMT EQU  ZBELL+3   ;TURN ON TRANSMIT LED
  196   481A   .   .   .   ZCLXMT EQU  ZSTXMT+3  ;TURN OFF TRANSMIT LED
  197   481D   .   .   .   ZSTJPR EQU  ZCLXMT+3  ;SET JUMPERS ESC SEQ ROUTINE
  198   4820   .   .   .   ZSTLKY EQU  ZSTJPR+3  ;SET LATCHING KEYS ROUTINE
  199   4823   .   .   .   ZALPCK EQU  ZSTLKY+3  ;ALPHA KEY ENTRY CHECK
  200   4826   .   .   .   ZNUMCK EQU  ZALPCK+3  ;NUMERIC KEY ENTRY CHECK
  201   0000   .   .   .   ;
  202   0000   .   .   .   ;  KEYBOARD CONSTANTS
  203   0000   .   .   .   ;
  204   4829   .   .   .   FRSALT EQU  ZNUMCK+3  ;INITIAL ALTERNATE CHAR SET
  205   482A   .   .   .   ALTOUT EQU  FRSALT+1  ;INITIAL ALTERNATE CHAR OUT
  206   0000   .   .   .   ;
  207   0000   .   .   .   ;  KEYBOARD CONTROL CALLS
  208   0000   .   .   .   ;
  209   0001   .   .   .   LOCKKB EQU  1        ;LOCK KEYBOARD
  210   0002   .   .   .   UNLKKB EQU  2        ;UNLOCK KEYBOARD
  211   0003   .   .   .   RPTKEY EQU  3        ;REPEAT LAST KEY HIT
  212   0004   .   .   .   STBLMD EQU  4        ;SET PERMANENT BLOCK MODE
  213   0005   .   .   .   STRTST EQU  5        ;START SELF-TEST
  214   0006   .   .   .   ENDTST EQU  6        ;END SELF-TEST
  215   0007   .   .   .   RSETKB EQU  7        ;RESET KEYBOARD
  216   0008   .   .   .   CKIOKY EQU  8        ;CHECK FOR I/O CONTROL KEY
  217   0009   .   .   .   STPRPT EQU  9        ;STOP KEY REPEAT
  218   000A   .   .   .   CKBRKY EQU  10       ;CHECK FOR BREAK KEY DOWN
  219   000B   .   .   .   SWCHAR EQU  11       ;SWITCH CHARACTER SET
  220   000C   .   .   .   SETFRN EQU  12       ;UPDATE FOREIGN MODE
  221   000D   .   .   .   STCHST EQU  13       ;SET FOREIGN OUTPUT MODE
  222   000E   .   .   .   FRNMD1 EQU  14       ;SET FOREIGN MODE 1
  223   000F   .   .   .   FRNMD2 EQU  15       ;SET FOREIGN MODE 2
  225   0000    .    .    .   ;**********************************************
  226   0000    .    .    .   ;
  227   0000    .    .    .   ;         DATACOM CONSTANTS
  228   0000    .    .    .   ;
  229   0000    .    .    .   ;**********************************************
  230   5000    .    .    .   ZDCBAS EQU  50000Q     ;DATACOM START ADDRESS
  231   5002    .    .    .   TRIGGR EQU  ZDCBAS+2   ;BLOCK TRANSFER TRIGGER
  232   5003    .    .    .   RECSEP EQU  TRIGGR+1   ;RECORD SEPARATOR CHARACTER
  233   5004    .    .    .   BLKTRM EQU  RECSEP+1   ;BLOCK TERMINATOR CHARACTER
  234   5005    .    .    .   DCJMSK EQU  BLKTRM+1   ;DATA COMM JUMPER MASK
  235   5006    .    .    .   DCJMS2 EQU  DCJMSK+1   ;DATA COMM JUMPER MASK #2
  236   0000    .    .    .   ;**********************************************
  237   0000    .    .    .   ;
  238   0000    .    .    .   ;         DATACOM ENTRY VECTOR POINTERS
  239   0000    .    .    .   ;
  240   0000    .    .    .   ;**********************************************
  241   5008    .    .    .   ZINIDC EQU  ZDCBAS+10Q   ;INITIALIZE DATACOM
  242   500B    .    .    .   ZIN2DC EQU  ZINIDC+3  ;INITIALIZATION CONTINUATOR
  243   500E    .    .    .   ZDCMON EQU  ZIN2DC+3  ;MONITORING ROUTINE
  244   5011    .    .    .   ZDCCTL EQU  ZDCMON+3  ;MISC CONTROL FUNCTIONS
  245   5014    .    .    .   ZDCTST EQU  ZDCCTL+3  ;SELF-TEST
  246   5017    .    .    .   ZGETDC EQU  ZDCTST+3  ;GET DC CHARACTER
  247   501A    .    .    .   ZPUTDC EQU  ZGETDC+3  ;PUT DC CHARACTER
  248   501D    .    .    .   ZGTBIN EQU  ZPUTDC+3  ;GET BINARY DC CHARACTER
  249   5020    .    .    .   ZSTBIN EQU  ZGTBIN+3  ;START BINARY OUTPUT
  250   5023    .    .    .   ZNDBIN EQU  ZSTBIN+3  ;END BINARY OUTPUT
  251   5026    .    .    .   ZDCINT EQU  ZNDBIN+3  ;DATACOM INTERRUPTS
  252   0000    .    .    .   ;**********************************************
  253   0000    .    .    .   ;
  254   0000    .    .    .   ;         DATACOM CONTROL CALL CODES
  255   0000    .    .    .   ;
  256   0000    .    .    .   ;**********************************************
  257   0000    .    .    .   CLRTRG EQU  0          ;CLEAR BLOCK TRANSFER TRIGGE
  258   0001    .    .    .   SETTRG EQU  1          ;SET BLOCK TRANSFER TRIGGER
  259   0002    .    .    .   RSETDC EQU  2          ;RESET DATACOM
  260   0003    .    .    .   SETREM EQU  3          ;SET REMOTE MODE
  261   0004    .    .    .   SETLCL EQU  4          ;SET LOCAL MODE
  262   0005    .    .    .   PUTBRK EQU  5          ;OUTPUT BREAK SIGNAL
  263   0006    .    .    .   DISCNT EQU  6          ;MODEM DISCONNECT
  264   0007    .    .    .   ENDBLK EQU  7          ;TERMINATE OUTPUT MESSAGE
  265   0008    .    .    .   SETMON EQU  8          ;ENTER MONITOR MODE
  266   0009    .    .    .   SETNRM EQU  9          ;ENTER NORMAL MODE
  267   000A    .    .    .   FSTBIN EQU  10         ;ENTER FAST BINARY OUT MODE
  268   000B    .    .    .   SNDATN EQU  11         ;SEND ATTENTION CODE
  269   000C    .    .    .   SNDFCT EQU  12         ;SEND FUNCTION DATA
  270   000D    .    .    .   PROMPT EQU  13         ;SEND PROMPT CODE
```

```
===========================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                     PAGE   7
===========================================================================
  272   0000     .   .   .   ;******************************
  273   0000     .   .   .   ; ALTERNATE I/O ENTRY VECTORS *
  274   0000     .   .   .   ;******************************
  275   6000     .   .   .   ALTORG EQU  60000Q    ;ALTERNATE I/O START ADDRESS
  276   6002     .   .   .   ZINIAL EQU  ALTORG+2  ;INITIALIZATION ROUTINE
  277   6005     .   .   .   ZIN2AL EQU  ZINIAL+3  ;INITIALIZATION CONTINUATOR
  278   6008     .   .   .   ZINTAL EQU  ZIN2AL+3  ;INTERRUPT PROCESSOR
  279   600B     .   .   .   ZMONAL EQU  ZINTAL+3  ;MONITORING ROUTINE
  280   600E     .   .   .   ZGETAL EQU  ZMONAL+3  ;INPUT ROUTINE
  281   6011     .   .   .   ZPUTAL EQU  ZGETAL+3  ;OUTPUT ROUTINE
  282   6014     .   .   .   ZCTLAL EQU  ZPUTAL+3  ;CONTROL ROUTINE
  283   6017     .   .   .   ZSTAAL EQU  ZCTLAL+3  ;STATUS ROUTINE
  284   601A     .   .   .   ZMSGAL EQU  ZSTAAL+3  ;ALTERNATE DEVICE NAME
 286   0000    .    .    .    ;**************************
 287   0000    .    .    .    ; ASCII CHARACTER EQUATES *
 288   0000    .    .    .    ;**************************
 289   0000    .    .    .    NULL    EQU   0Q          ;NULL
 290   000A    .    .    .    LF      EQU   12Q         ;LINE FEED
 291   000C    .    .    .    FF      EQU   14Q         ;FORM FEED
 292   000D    .    .    .    CR      EQU   15Q         ;RETURN
 293   000E    .    .    .    SO      EQU 016Q
 294   000F    .    .    .    SI      EQU 017Q
 295   0012    .    .    .    DC2     EQU   22Q         ;DEVICE CONTROL 2
 296   0013    .    .    .    DC3     EQU   23Q         ;DEVICE CONTROL 3
 297   001B    .    .    .    ESC     EQU   33Q         ;ESCAPE
 298   0020    .    .    .    CTLLIM  EQU   40Q         ;CONTROL CODE UPPER LIMIT
 299   0020    .    .    .    ABLNK   EQU 040Q        ;ASCII BLANK
 300   0026    .    .    .    AMPSND  EQU   46Q         ;(&) - AMPERSAND
 301   0027    .    .    .    QUOTE   EQU   47Q         ;(') - SINGLE QUOTE
 302   0029    .    .    .    ARPARN  EQU   51Q         ;[)] - RIGHT PARENTHESIS
 303   002B    .    .    .    PLUS    EQU   53Q         ;PLUS SIGN
 304   002C    .    .    .    COMMA   EQU   54Q         ;COMMA
 305   002D    .    .    .    MINUS   EQU   55Q         ;MINUS SIGN
 306   002E    .    .    .    PERIOD  EQU   56Q         ;(.) - PERIOD
 307   002F    .    .    .    SLANT   EQU   57Q         ;(/) - SLANT
 308   0030    .    .    .    ZERO    EQU   60Q         ;ASCII ZERO
 309   0032    .    .    .    TWO     EQU   62Q         ;ASCII TWO
 310   0033    .    .    .    THREE   EQU   63Q         ;ASCII THREE
 311   0034    .    .    .    FOUR    EQU   64Q         ;ASCII FOUR
 312   0035    .    .    .    FIVE    EQU   65Q         ;ASCII FIVE
 313   0036    .    .    .    SIX     EQU   66Q         ;ASCII SIX
 314   0037    .    .    .    SEVEN   EQU   67Q         ;ASCII SEVEN
 315   0000    .    .    .    ;
 316   0040    .    .    .    ATSIGN  EQU   100Q        ;"AT" SIGN (@)
 317   0041    .    .    .    A       EQU   101Q        ;UPPER CASE A
 318   0043    .    .    .    C       EQU   103Q        ;UPPER CASE C
 319   0044    .    .    .    D       EQU   104Q        ;UPPER CASE D
 320   0046    .    .    .    F       EQU   106Q        ;UPPER CASE F
 321   0048    .    .    .    H       EQU   110Q        ;UPPER CASE H
 322   004C    .    .    .    L       EQU   114Q        ;UPPER CASE L
 323   004E    .    .    .    N       EQU   116Q        ;UPPER CASE N
 324   0050    .    .    .    P       EQU   120Q        ;UPPER CASE P
 325   0052    .    .    .    R       EQU   122Q        ;UPPER CASE R
 326   0053    .    .    .    S       EQU   123Q        ;UPPER CASE S
 327   0054    .    .    .    T       EQU   124Q        ;UPPER CASE T
 328   0055    .    .    .    U       EQU   125Q        ;UPPER CASE U
 329   0059    .    .    .    Y       EQU   131Q        ;UPPER CASE Y
 330   005A    .    .    .    Z       EQU   132Q        ;UPPER CASE Z
 331   005B    .    .    .    LFTBKT  EQU   133Q        ;LEFT BRACKET
 332   005C    .    .    .    ABCKSL  EQU   134Q        ;(\) - BACK SLANT
```

```
===============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                          PAGE   9
===============================================================================
  334   0000    .   .   .   ;**********************
  335   0000    .   .   .   ; LOWER CASE EQUATES *
  336   0000    .   .   .   ;**********************
  337   0061    .   .   .   SMALLA EQU  141Q       ;LOWER CASE A
  338   0063    .   .   .   ALCC   EQU 143Q      ;ASCII LOWER CASE C
  339   0064    .   .   .   SMALLD EQU  144Q       ;LOWER CASE D
  340   0066    .   .   .   SMALLF EQU  146Q       ;LOWER CASE F
  341   0069    .   .   .   SMALLI EQU  151Q       ;LOWER CASE I
  342   006B    .   .   .   SMALLK EQU  153Q       ;LOWER CASE K
  343   0070    .   .   .   SMALLP EQU  160Q       ;LOWER CASE P
  344   0078    .   .   .   SMALLX EQU  170Q       ;LOWER CASE X
  345   007B    .   .   .   LFTBRC EQU  173Q       ;LEFT BRACE
  346   007C    .   .   .   VRTBAR EQU  174Q       ;VERTICAL BAR
  347   007F    .   .   .   ADEL   EQU  177Q       ;DELETE (RUBOUT)
```

```
==========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
==========================================================================
  349   0000   .   .   .    ;************************
  350   0000   .   .   .    ; DISPLAY FLAGS EQUATES *
  351   0000   .   .   .    ;************************
  352   00BF   .   .   .    ENHLIM EQU  277Q       ;MAXIMUM ENHANCEMENT CODE
  353   00C0   .   .   .    STPR   EQU  300Q       ;START PROTECTED FIELD
  354   00C1   .   .   .    ENDPR  EQU  301Q       ;END PROTECTED FIELD
  355   00C2   .   .   .    XMONLY EQU  302Q       ;START TRANSMIT-ONLY FIELD
  356   00C3   .   .   .    FILL   EQU  303Q       ;EOL FILL CHARACTER
  357   00C4   .   .   .    STPFLG EQU  304Q       ;NON-DISPLAYING TERMINATOR
  358   00C5   .   .   .    ALPHA  EQU  305Q       ;ALPHABETIC ONLY
  359   00C6   .   .   .    NUMBER EQU  306Q       ;NUMERIC ONLY
  360   00C7   .   .   .    ALPHNM EQU  307Q       ;ALPHANUMERIC FIELD
  361   00C8   .   .   .    SFKYAT EQU  310Q       ;SOFT KEY ATTRIBUTE FIELD
  362   0000   .   .   .    ;
  363   00C4   .   .   .    FLDSEP EQU  304Q       ;FIELD SEPARATOR FOR I/O BUF
  364   00CC   .   .   .    EOL    EQU 314Q
  365   00CE   .   .   .    EOP    EQU 316Q
  366   00D0   .   .   .    LNKLIM EQU 320Q     ;LOWEST VALUE FOR A LINK
  367   0800   .   .   .    NUM2K  EQU  4000Q      ;NUMBER 2048 (2K)
  368   8000   .   .   .    B15    EQU  100000Q    ;BIT 15
  369   00C3   .   .   .    JMP    EQU  303Q       ;JUMP INSTRUCTION CODE
  370   00C9   .   .   .    RET    EQU  311Q       ;RETURN INSTRUCTION CODE
  371   0000   .   .   .    ;************************
  372   0000   .   .   .    ; MISCELLANEOUS EQUATES *
  373   0000   .   .   .    ;************************
  374   0017   .   .   .    MAXROW EQU  23         ;MAXIMUM ROW NUMBER
  375   004F   .   .   .    MAXCOL EQU  79         ;MAXIMUM COLUMN NUMBER
  376   0010   .   .   .    SFTEND EQU  16         ;LAST SOFT KEY DEFINITION RO
  377   0008   .   .   .    BELLIM EQU  8          ;SPACE FROM RHTMGN FOR BELL
  378   000F   .   .   .    BLKSM  EQU  17Q        ;BLOCK SIZE MASK
  379   0010   .   .   .    BLKSZ  EQU  16         ;BLOCK SIZE
  380   0008   .   .   .    IOERRB EQU  10Q        ;I/O ERROR STATUS BIT
  381   0001   .   .   .    REXMIT EQU  1Q         ;RE-TRANSMIT I/O FLAG
  382   0002   .   .   .    BINXMT EQU  2          ;SEND BINARY DATA
  383   0032   .   .   .    SFTDLY EQU  50         ;SOFT RESET PERIOD - .50 SEC
  384   0080   .   .   .    NOSIGN EQU  200Q       ;NO SIGN FLAG FOR INPUT DATA
```

```
=====================================================================================
                                                                        PAGE  11
ITEM    LOC    OBJECT CODE   SOURCE STATEMENTS
=====================================================================================
 386    0000     .    .    .    ;*********************
 387    0000     .    .    .    ; I/O MODULE EQUATES *
 388    0000     .    .    .    ;*********************
 389    0000     .    .    .    RESET  EQU   0Q         ;RESET TERMINAL VECTOR
 390    0001     .    .    .    RSTJMP EQU   1Q         ;VECTOR FOR RESTART "PCHL"
 391    0070     .    .    .    PROCSR EQU   160Q       ;PROCESSOR "OUT" PORT
 392    0080     .    .    .    IOBASE EQU 200Q      ;I/O ADDRESS MSB'S
 393    0000     .    .    .    ;
 394    0000     .    .    .    ;  KEYBOARD
 395    0000     .    .    .    ;
 396    8300     .    .    .    IOKB   EQU   (3Q+IOBASE)*256;MODULE 11 BASE ADDRESS
 397    8380     .    .    .    IOKBCO EQU   IOKB+200Q  ;RESET KEY CONTROL
 398    0002     .    .    .    RSTON  EQU   2Q         ;RESET ON
 399    0004     .    .    .    RSTOFF EQU   4Q         ;RESET OFF
 400    0008     .    .    .    NMFCTK EQU   8          ;NUMBER OF FUNCTION KEYS
 401    0000     .    .    .    ;
 402    0000     .    .    .    ;  CURSOR CONTROL
 403    0000     .    .    .    ;
 404    8700     .    .    .    IODISP EQU   (7Q+IOBASE)*256;MODULE 13 BASE ADDRESS
 405    8700     .    .    .    IOCRCL EQU   IODISP+0  ;CURSOR COLUMN ADDRESS
 406    8720     .    .    .    IOCRRW EQU   IODISP+40Q  ;CURSOR ROW ADDRESS
 407    0020     .    .    .    MAYEOP EQU   40Q        ;DMA ON, EOP IF DMA ROW = RO
 408    0040     .    .    .    MAYEOL EQU   100Q       ;DMA OFF, SKIP EOP IF ROWS =
 409    0060     .    .    .    DMAOFF EQU   140Q       ;DMA OFF
 410    0080     .    .    .    CRTOFF EQU   200Q       ;DISPLAY OFF
 411    0082     .    .    .    INVRS  EQU   202Q       ;INVERSE VIDEO ON
 412    0080     .    .    .    NORMAL EQU   200Q       ;NORMAL VIDEO ON
 413    0000     .    .    .    ;
 414    0000     .    .    .    ;  CARTRIDGE TAPE
 415    0000     .    .    .    ;
 416    8B00     .    .    .    IOCTU  EQU   (13Q+IOBASE*256);MODULE 15 BASE ADDRES
 417    8B00     .    .    .    IOCTCO EQU   IOCTU+0Q  ;COMMAND TO CTU
 418    8B00     .    .    .    IOCTSI EQU   IOCTU+0Q  ;STATUS FROM CTU
 419    8B20     .    .    .    IOCTDO EQU   IOCTU+40Q  ;DATA TO CTU
 420    8B20     .    .    .    IOCTDI EQU   IOCTU+40Q  ;DATA FROM CTU
```

```
  422    0000    .    .    .    ;
  423    0000    .    .    .    ;  9866 PRINTER
  424    0000    .    .    .    ;
  425    8D00    .    .    .    IOPTR1 EQU  (15Q+IOBASE)*256;MODULE 16 BASE ADDRES
  426    8D20    .    .    .    PTROT1 EQU  IOPTR1+40Q  ;PRINTER DATA OUT
  427    8D00    .    .    .    PTRST1 EQU  IOPTR1+0Q  ;PRINTER STATUS IN
  428    8D02    .    .    .    PTRCL1 EQU  IOPTR1+2Q  ;PRINTER CLEAR
  429    0000    .    .    .    ;
  430    0000    .    .    .    ;  RS-232 PRINTER
  431    0000    .    .    .   .;
  432    8500    .    .    .    IOPTR2 EQU  (5Q+IOBASE)*256;MODULE 12 BASE ADDRESS
  433    8540    .    .    .    PTROT2 EQU  IOPTR2+100Q  ;INTERFACE CONTROL OUT
  434    8520    .    .    .    PTRST2 EQU  IOPTR2+40Q  ;PRINTER STATUS IN
  435    8560    .    .    .    PTRDA2 EQU  IOPTR2+140Q  ;PRINTER DATA OUT
  436    8540    .    .    .    PTRCF2 EQU  IOPTR2+100Q  ;OPTION JUMPERS IN
```

```
 438   0000   .    .    .   ;******************
 439   0000   .    .    .   ; PRINTER EQUATES *
 440   0000   .    .    .   ;******************
 441   0000   .    .    .   ;
 442   0000   .    .    .   ;  RS-232 OPTION STRAPS
 443   0000   .    .    .   ;
 444   0000   .    .    .   ;      BITS 2-0    MEANING IF SET
 445   0000   .    .    .   ;      000         EXT BAUD RATE
 446   0000   .    .    .   ;      001          110  "
 447   0000   .    .    .   ;      010          150  "
 448   0000   .    .    .   ;      011          300  "
 449   0000   .    .    .   ;      100         1200  "
 450   0000   .    .    .   ;      101         2400  "
 451   0000   .    .    .   ;      110         4800  "
 452   0000   .    .    .   ;      111         9600  "
 453   0000   .    .    .   ;
 454   0000   .    .    .   ;      BIT 3       PARITY SELECT
 455   0000   .    .    .   ;        1         EVEN
 456   0000   .    .    .   ;        0         ODD
 457   0000   .    .    .   ;
 458   0000   .    .    .   ;      BIT 4       PARITY INHIBIT
 459   0000   .    .    .   ;        1         NO PARITY
 460   0000   .    .    .   ;        0         PARITY
 461   0000   .    .    .   ;      BITS 7-5    # OF FILLS
 462   0000   .    .    .   ;      000         HANDSHAKE DEVICE
 463   0000   .    .    .   ;      001         8
 464   0000   .    .    .   ;      010         16
 465   0000   .    .    .   ;      011         24
 466   0000   .    .    .   ;      100         32
 467   0000   .    .    .   ;      101         40
 468   0000   .    .    .   ;      110         48
 469   0000   .    .    .   ;      111         56
 470   0000   .    .    .   ;*****************
 471   0000   .    .    .   ; DRIVER EQUATES *
 472   0000   .    .    .   ;*****************
 473   05DC   .    .    .   PTDLY  EQU  1500      ;15 SECOND PRINTER TIME OUT
 474   0000   .    .    .   ;************************
 475   0000   .    .    .   ; 9866 PRINTER EQUATES *
 476   0000   .    .    .   ;************************
 477   0001   .    .    .   PTRDY1 EQU 1          ;PRINTER READY
 478   0080   .    .    .   PTRPO1 EQU 200Q       ;PRINTER OUT OF PAPER
 479   0000   .    .    .   ;**************************
 480   0000   .    .    .   ; RS-232 PRINTER EQUATES *
 481   0000   .    .    .   ;**************************
 482   0002   .    .    .   PTRDY2 EQU 2          ;PRINTER READY MASK
 483   0040   .    .    .   PTRSB2 EQU 100Q       ;RS-232 SB LINE STROBE
 484   0020   .    .    .   PTROL2 EQU 40Q        ;PRINTER READY MASK
 485   00E0   .    .    .   PTRHD2 EQU 340Q       ;RS-232 HANDSHAKE PROTOCOL
 486   001F   .    .    .   PTRBD2 EQU 370        ;PARITY AND BAUD RATE MASK
```

```
  488   0000    .   .   .   ;****************************
  489   0000    .   .   .   ; VARIABLE SPACE ALLOCATION *
  490   0000    .   .   .   ;****************************
  491   FBFF    .   .   .   DSPLIM EQU  175777Q   ;DISPLAY UPPER LIMIT
  492   00D0    .   .   .   LWDSP  EQU  150000Q/256  ;DISPLAY LOWER LIMIT
  493   FC00    .   .   .   IOBUF  EQU  176000Q
  494   00FC    .   .   .   IOBUFH EQU  IOBUF/256
  495   0000    .   .   .   IOBUFL EQU  -IOBUFH*256+IOBUF
  496   FC00    .   .   .   IOBUF1 EQU  176000Q
  497   FD00    .   .   .   IOBUF2 EQU  176400Q
  498   FE4F    .   .   .   DSPSTR EQU  177000Q+79  ;MESSAGE BUFFER
  499   0100    .   .   .   PTRBLN EQU  256        ;PRINTER INPUT BUFFER SIZE
  500   0000    .   .   .   ;***************************
  501   0000    .   .   .   ; OPERATING SYSTEM STORAGE *
  502   0000    .   .   .   ;***************************
  503   9160    .   .   .   STACK  EQU  FSTRAM+140Q  ;STACK AREA (96 BYTES)
  504   FFD0    .   .   .   OPSTOR EQU  177720Q   ;VARIABLES STORAGE AREA
  505   00FF    .   .   .   BASEH  EQU  OPSTOR/256  ;MSB OF DATA PAGE ADDRESSE
  506   FF00    .   .   .   BASE   EQU  BASEH*256  ;DATA PAGE BASE ADDRESS
  507   00FE    .   .   .   BASEH2 EQU  BASEH-1   ;BASE VALUES FOR SECOND PAGE
  508   FE00    .   .   .   BASE2  EQU  BASEH2*256  ;OF VARIABLES SPACE
  509   0000    .   .   .   ;***************************
  510   0000    .   .   .   ; VARIABLE SUBROUTINE CALL *
  511   0000    .   .   .   ;***************************
  512   FFCD    .   .   .   ECONTF EQU OPSTOR-3 ;JUMP SUBROUTINE
  513   FFCE    .   .   .   CNTFAD EQU  ECONTF+1  ;CHARACTER FUNCTION ADDRESS
```

```
=================================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                              PAGE  15
=================================================================================
  515    0000    .   .   .   ;*********************************************
  516    0000    .   .   .   ; NORMAL/SOFT KEY SWAPPED DISPLAY PARAMETERS *
  517    0000    .   .   .   ;*********************************************
  518    FFCB    .   .   .   TOPLIN EQU  ECONTF-2  ;LSB PART OF NEXT LINE
  519    0000    .   .   .   ;                       POINTER IN TOP DISPLAY
  520    0000    .   .   .   ;                       LINE
  521    FFC9    .   .   .   LSTLIN EQU  TOPLIN-2  ;POINTER TO LSB PART OF
  522    0000    .   .   .   ;                       NEXT LINE POINTER IN
  523    0000    .   .   .   ;                       LAST LINE PROCESSED
  524    FFC8    .   .   .   LSTCOL EQU  LSTLIN-1  ;COLUMN AND ROW POSITION OF
  525    FFC7    .   .   .   LSTROW EQU  LSTCOL-1    ;LAST CHARACTER PROCESSED
  526    0000    .   .   .   ;                       (CORRESPONDS TO CHARACTER
  527    0000    .   .   .   ;                          GIVEN BY "CURADR")
  528    FFC6    .   .   .   LSTDCD EQU  LSTROW-1  ;LAST DISPLAY CODE USED
  529    FFC5    .   .   .   LSTFMT EQU  LSTDCD-1  ;LAST FORMAT CONTROL USED
  530    FFC3    .   .   .   CURADR EQU  LSTFMT-2  ;ADDRESS OF LAST CHARACTER
  531    0000    .   .   .   ;                       PROCESSED
  532    FFC2    .   .   .   PROFLD EQU  CURADR-1  ;PROTECT STATE OF (CURADR)
  533    0000    .   .   .   ;                       = -1, PROTECTED
  534    0000    .   .   .   ;                       # -1, NOT PROTECTED
  535    0000    .   .   .   ;************************
  536    0000    .   .   .   ; CURRENT CURSOR VALUES *
  537    0000    .   .   .   ;************************
  538    FFC1    .   .   .   CURCOL EQU  PROFLD-1  ;CURRENT COLUMN AND ROW
  539    FFC0    .   .   .   CURROW EQU  CURCOL-1    ;POSITION OF CURSOR
  540    FFBF    .   .   .   LFTMGN EQU  CURROW-1  ;LEFT MARGIN SETTING
  541    FFBE    .   .   .   RHTMGN EQU  LFTMGN-1  ;RIGHT MARGIN SETTING
  542    000F    .   .   .   NUMSWP EQU  ECONTF-RHTMGN  ;# OF SWAP VARIABLES
  543    FFAF    .   .   .   SWPSTR EQU  RHTMGN-NUMSWP  ;SWAP BUFFER
  544    FFAE    .   .   .   DSPTYP EQU  SWPSTR-1  ;DISPLAY CURRENTLY ENABLED
  545    0000    .   .   .   ;                       0 = NORMAL DISPLAY
  546    0000    .   .   .   ;                      -1 = SOFT KEY DISPLAY
  547    0000    .   .   .   ;*****************************************
  548    0000    .   .   .   ; FIXED DISPLAY PARAMETERS (NOT SWAPPED) *
  549    0000    .   .   .   ;*****************************************
  550    FFAC    .   .   .   FRBLKS EQU  DSPTYP-2  ;FREE BLOCKS LIST HEAD
  551    FFAA    .   .   .   DSPBGN EQU  FRBLKS-2  ;LOW ADDRESS OF DISPLAY AREA
  552    FFA8    .   .   .   DSPEND EQU  DSPBGN-2  ;HIGH ADDR OF DISPLAY AREA
  553    FFA6    .   .   .   SFTKYS EQU  DSPEND-2  ;SOFT KEY DISPLAY START ADDR
  554    FFA4    .   .   .   CURFKY EQU  SFTKYS-2  ;CURRENT FUNCTION KEY CHAR
  555    FFA3    .   .   .   TLINO  EQU  CURFKY-1  ;TOP LINE ABSOLUTE ROW NUMBE
  556    FFA1    .   .   .   LLINE  EQU  TLINO-2   ;LAST DISPLAY LINE START ADD
  557    FF9F    .   .   .   FLINE  EQU  LLINE-2   ;POINTER TO LSB PART OF NEXT
  558    0000    .   .   .   ;                       LINE POINTER IN FIRST
  559    0000    .   .   .   ;                       LINE OF NORMAL DISPLAY
```

```
=========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                     PAGE  16
=========================================================================
  561   0000    .    .    .   ;********************
  562   0000    .    .    .   ; SCRATCH VARIABLES *
  563   0000    .    .    .   ;********************
  564   FF9E    .    .    .   TEMP1   EQU   FLINE-1
  565   FF9D    .    .    .   TEMP    EQU   TEMP1-1     ;TEMPORARY STORAGE
  566   FF9C    .    .    .   CHARIN  EQU   TEMP-1      ;CHARACTER FROM KEYBOARD
  567   FF9B    .    .    .   NCHAR   EQU   CHARIN-1   ;NUMBER OF CHARS TO BE ADDED
  568   FF9A    .    .    .   NROWS   EQU NCHAR-1   ;NO. OF ROWS TO BE ADDED
  569   FF99    .    .    .   NBLKS   EQU NROWS-1   ;NO. OF BLOCKS TO BE ADDED
  570   FF98    .    .    .   CHSAV   EQU NBLKS-1   ;SAVE AREA FOR CHAR
  571   0000    .    .    .   ;                      PRECEDING LINK
  572   FF96    .    .    .   LNKSAV EQU CHSAV-2   ;LINK SAVE AREA
  573   FF94    .    .    .   EOLADR EQU LNKSAV-2 ;ADDR OF LAST EOL
  574   FF92    .    .    .   FRSTBL EQU EOLADR-2 ;FIRST BLOCK IN DISPL1
  575   FF91    .    .    .   BLKFIL EQU FRSTBL-1 ;FILL FLAG FOR FNDCHR
  576   FF90    .    .    .   EOLMV   EQU BLKFIL-1 ;FLAG FOR EOLMOV
  577   FF8F    .    .    .   FILCHR EQU EOLMV-1   ;FILL CHAR SAVE FOR GTBLK
  578   CFFF    .    .    .   BFSPCE EQU   147777Q    ;UPPER LIMIT OF BUFFER
  579   0080    .    .    .   LWBUF   EQU   130000Q/256  ;LOWER LIMIT
  580   FF8D    .    .    .   BUFBGN EQU   FILCHR-2   ;LOW ADDR OF NON-DISPLY BUFF
  581   FF8B    .    .    .   BUFEND EQU   BUFBGN-2   ;HIGH ADDR FOR BUFFER
  582   0000    .    .    .   ;******************************************
  583   0000    .    .    .   ;  STORAGE FOR CHARACTERS TO BE STORED  *
  584   0000    .    .    .   ;******************************************
  585   FF8A    .    .    .   FMTCTL EQU   BUFEND-1   ;FORMAT CONTROL TO BE ENTERE
  586   FF89    .    .    .   DCHAR   EQU   FMTCTL-1   ;NEXT CHAR TO BE DISPLAYED
  587   FF88    .    .    .   CHAR    EQU   DCHAR-1    ;CURRENT CHAR BEING PROCESSE
  588   FF86    .    .    .   CHKRTN EQU   CHAR-2     ;CURRENT TYPE CHECK ROUTINE
  589   FF85    .    .    .   TMPCOL EQU   CHKRTN-1   ;COLUMN # STORAGE FOR RCADDR
  590   0000    .    .    .   ;**********************************
  591   0000    .    .    .   ; STORAGE FOR CURSOR POSITIONING *
  592   0000    .    .    .   ;**********************************
  593   FF84    .    .    .   COUNT   EQU   TMPCOL-1   ;NUMBER OF BYTES TO FILL
  594   FF83    .    .    .   NMROLL EQU   COUNT-1    ;NUMBER OF LINES TO ROLL
  595   FF82    .    .    .   ROLLCT EQU   NMROLL-1   ;ROLL COUNTER
  596   0000    .    .    .   ;
  597   FFDB    .    .    .   NEWCOL EQU   PARM1      ;NEW COLUMN NUMBER
  598   FFDA    .    .    .   NEWROW EQU   PARM2      ;NEW ABSOLUTE ROW NUMBER
  599   FFD9    .    .    .   SCRNRW EQU   PARM3      ;NEW SCREEN ROW SETTING
```

```
==================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                           PAGE  17
==================================================================================
 601    0000   .    .    .   ;***********************
 602    0000   .    .    .   ; HORIZONTAL TAB TABLE *
 603    0000   .    .    .   ;***********************
 604    000A   .    .    .   HTBLEN EQU  10        ;TABLE LENGTH (= 10 X 8)
 605    FF78   .    .    .   HTBTBL EQU  ROLLCT-HTBLEN
 606    0000   .    .    .   ;***********************
 607    0000   .    .    .   ; DISPLAY SEND STORAGE *
 608    0000   .    .    .   ;***********************
 609    FF77   .    .    .   CDSPEN EQU HTBTBL-1 ;CURRENT ENHANCEMENT IN
 610    FF76   .    .    .   ENHOUT EQU CDSPEN-1 ;LAST ENHANCEMENT OUT
 611    FF75   .    .    .   CALTST EQU ENHOUT-1 ;CURRENT ALTERNATE SET OUT
 612    FF73   .    .    .   GETADR EQU  CALTST-2  ;CURRENT CHARACTER ADDRESS
 613    0000   .    .    .   ;****************************
 614    0000   .    .    .   ; FLAGS AND TABLE POINTERS *
 615    0000   .    .    .   ;****************************
 616    FF72   .    .    .   CHRSET EQU  GETADR-1  ;CURRENT ALTERNATE CHAR SET
 617    FF71   .    .    .   KBFCTK EQU  CHRSET-1  ;KEYBOARD FUNCTION CODE
 618    0000   .    .    .   ;*************************************************
 619    FF70   .    .    .   MFLGS  EQU  KBFCTK-1  ;BLOCK TRANSFER PENDING FLAG
 620    0000   .    .    .   ;*************************************************
 621    0100   .    .    .   SDC2   EQU  1Q*256    ;DC2 PENDING
 622    0200   .    .    .   SSTAT  EQU  2Q*256    ;TERMINAL STATUS PENDING
 623    0400   .    .    .   SSTAT2 EQU  4Q*256    ;TERMINAL STATUS 2 PENDING
 624    0800   .    .    .   SDVST  EQU  10Q*256   ;DEVICE STATUS PENDING
 625    1000   .    .    .   SCRSEN EQU  20Q*256   ;CURSOR SENSE PENDING
 626    2000   .    .    .   SFCTKY EQU  40Q*256   ;FUNCTION KEY PENDING
 627    4000   .    .    .   SENTER EQU  100Q*256  ;DISPLAY SEND PENDING
 628    8000   .    .    .   SDVDUN EQU  200Q*256  ;DEVICE DONE PENDING
 629    0000   .    .    .   ;****************************************
 630    FF6F   .    .    .   MFLGS2 EQU  MFLGS-1   ;MAIN CODE MODE FLAGS
 631    0000   .    .    .   ;****************************************
 632    0001   .    .    .   SDVREC EQU  1Q        ;DEVICE RECORD PENDING
 633    0002   .    .    .   SBINRY EQU  2Q        ;BINARY RECORD PENDING
 634    0004   .    .    .   RELSNS EQU  4Q        ;RELATIVE CURSOR SENSE
 635    0008   .    .    .   ESCINP EQU 10Q       ;ESC RECEIVED IN BLOCK MODE
 636    0010   .    .    .   FRSOUT EQU  20Q       ;FIRST SOFT KEY DATA OUT
 637    0020   .    .    .   WRPDEL EQU  40Q       ;DELETE CHAR W/ WRAP AROUND
 638    0040   .    .    .   WRPFLG EQU  100Q      ;LINE WRAP AROUND OCCURRED
 639    0080   .    .    .   NWRWST EQU  200Q      ;NEW ABSOLUTE ROW SET
 640    0000   .    .    .   ;***************************************
 641    FF6E   .    .    .   DFLGS  EQU  MFLGS2-1  ;DATA TRANSFER FLAGS
 642    0000   .    .    .   ;***************************************
 643    0001   .    .    .   SDACOM EQU 001Q      ;DATACOM/KEYBOARD
 644    0002   .    .    .   CNTXFR EQU  2Q        ;CONTINUE BUFFER TO DATA COM
 645    0004   .    .    .   NOSEND EQU  4Q        ;NO DISPLAY DATA TO SEND
 646    0008   .    .    .   SKPTRM EQU  10Q       ;SKIP BLOCK TERMINATOR
 647    0010   .    .    .   FCTK2D EQU  20Q       ;FUNCTION KEY TO DISPLAY
 648    0040   .    .    .   KBDLOK EQU  100Q      ;KB LOCKED BY ESCAPE SEQUENC
 649    0080   .    .    .   XBF2DS EQU  200Q      ;I/O BUFFER TO DISPLAY MODE
```

```
==========================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS
==========================================================================
  651   0000    .    .    .    ;*********************************************
  652   FF6D    .    .    .    TRMFCT EQU  DFLGS-1    ;NON-DISPLAYING TERMINATOR
  653   0000    .    .    .    ;*********************************************
  654   FFFF    .    .    .    STPXFR EQU  -1         ;TERMINATE TRANSFER
  655   0000    .    .    .    DELTRM EQU  0          ;DELETE TERMINATOR
  656   0001    .    .    .    IGNTRM EQU  1          ;IGNORE TERMINATOR
  657   0000    .    .    .    ;*****************************************
  658   FF6C    .    .    .    SPOWL  EQU  TRMFCT-1   ;SPACE OVERWRITE LATCH
  659   0000    .    .    .    ;*****************************************
  660   0020    .    .    .    SPOWON EQU  40Q        ;SPOW LATCH ON
  661   00FF    .    .    .    SPOWOF EQU  377Q       ;SPOW LATCH OFF
  662   0000    .    .    .    ;************************************************
  663   FF6B    .    .    .    MLKROW EQU  SPOWL-1    ;MEMORY LOCK ROW
  664   FF6A    .    .    .    MLKFLG EQU  MLKROW-1   ;MEMORY LOCK FLAG
  665   FF69    .    .    .    LCHAR  EQU  MLKFLG-1   ;LAST CHARACTER PROCESSED
  666   FF68    .    .    .    TCHAR  EQU  LCHAR-1    ;CURRENT TEST PATTERN CHAR
  667   FF67    .    .    .    CRAFLG EQU  TCHAR-1    ;CURSOR ADVANCE FLAG
  668   0000    .    .    .    ;*****************************
  669   0000    .    .    .    ; POINTERS FOR BINARY LOADER *
  670   0000    .    .    .    ;*****************************
  671   FFD5    .    .    .    LADDR  EQU  PARM6      ;BYTE ADDRESS PARAMETER
  672   FFDE    .    .    .    LDATA  EQU  IODATA     ;INPUT DATA ACCUMULATOR
  673   FFD7    .    .    .    LCHKSM EQU  PARM5      ;16-BIT CHECKSUM
```

```
====================================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS
====================================================================================
 675   0000   .    .    .   ;V*V*V*V*V*V*V*V*V*V*V*V*V*V*V*V*V*V*V*V
 676   0000   .    .    .   ;
 677   0000   .    .    .   ;  CTU/IO EQUATES - 4/11/76 - 2255 HOURS
 678   0000   .    .    .   ;
 679   0000   .    .    .   ; TAPE DISTANCE MEASUREMENT
 680   0000   .    .    .   ;==========================
 681   0000   .    .    .   ;
 682   0000   .    .    .   ; AS OF 3/1/75, .017125" OF TAPE MOTION IS
 683   0000   .    .    .   ; EQUIVALENT TO 1 TACH EDGE.  THE COUNT IS
 684   0000   .    .    .   ; IN ERROR WHEN STARTING OR STOPPING BY
 685   0000   .    .    .   ; 1 TACH EDGE (STOPPING IN A GAP MAY CAUSE
 686   0000   .    .    .   ; AN ERROR OF TWO TACH EDGES).
 687   0000   .    .    .   ;
 688   0000   .    .    .   ;*******************************
 689   FF66   .    .    .   CTSTAT EQU  CRAFLG-1  ;CTU STATUS
 690   0000   .    .    .   ;*******************************
 691   0080   .    .    .   TKI    EQU  200Q       ;TACH INTERRUPT
 692   0040   .    .    .   RDY    EQU 100Q     ;BYTE READY
 693   0020   .    .    .   GAP    EQU 40Q
 694   0010   .    .    .   HOL    EQU 20Q      ;TAPE HOLE
 695   0008   .    .    .   TAK    EQU 10Q      ;TACH (58.4 EDGES/IN)
 696   0004   .    .    .   RIP    EQU 4Q       ;RECORD IN PROGRESS
 697   0002   .    .    .   CIR    EQU  2Q        ;RIGHT CARTRIDGE INSERTED
 698   0001   .    .    .   CIL    EQU  1Q        ;LEFT CARTRIDGE INSERTED
 699   0000   .    .    .   ;********************************
 700   FF65   .    .    .   IOFLGS EQU  CTSTAT-1  ;I/O FLAGS 1
 701   0000   .    .    .   ;********************************
 702   0001   .    .    .   RDWOWT EQU  1Q        ;READ WITHOUT WAIT MODE
 703   0002   .    .    .   USREAD EQU  2Q        ;READ KEY INITIATED READ
 704   0004   .    .    .   FILRED EQU  4Q        ;FILE READ
 705   0008   .    .    .   RECRWD EQU  10Q       ;RECORD DISPLAY AND REWIND
 706   0000   .    .    .   ;                       OLD OUTPUT CTU (LOGGING)
 707   0010   .    .    .   RECINI EQU  20Q       ;START "RECORD" MODE
 708   0020   .    .    .   RECPGE EQU  40Q       ;FILE COPY FROM DISPLAY -
 709   0000   .    .    .   ;                       INHIBIT ROLL UP
 710   0080   .    .    .   VERIFY EQU  200Q      ;"CTU2BF" PERFORMS VERIFY
 711   0000   .    .    .   ;********************************
 712   FF64   .    .    .   IOFLG2 EQU  IOFLGS-1  ;I/O FLAGS 2
 713   0000   .    .    .   ;********************************
 714   0001   .    .    .   EXTB2D EQU  1Q        ;EXTERNAL BUFFER TO DATA COM
 715   0020   .    .    .   XDS2BF EQU  40Q       ;TRANSFER DISPLAY TO BUFFER
 716   0040   .    .    .   DSPB1M EQU  100Q      ;BOTTOM OF DISPLAY REACHED
 717   0080   .    .    .   ENDDSP EQU  200Q      ;END OF DISPLAY REACHED
```

```
========================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                               PAGE  20
========================================================================================
  719   0000   .   .   .    ;**********************************
  720   FF63   .   .   .    UNITO   EQU  IOFLG2-1  ;UNIT STATUS
  721   0000   .   .   .    ;**********************************
  722   0001   .   .   .    LPM     EQU    1Q       ;TAPE AT OR BEFORE LOAD POIN
  723   0002   .   .   .    LSTFWD  EQU    2Q       ;TAPE LAST MOVED FORWARD
  724   0004   .   .   .    FPS     EQU    4Q       ;TAPE WRITE PROTECTED
  725   0008   .   .   .    CMDEXC  EQU   10Q       ;SUCCESSFUL COMMAND EXECUTIO
  726   0010   .   .   .    DBLHOL  EQU   20Q       ;DOUBLE HOLE FOUND
  727   0020   .   .   .    BOT     EQU   40Q       ;TAPE PAST BOT HOLES
  728   0040   .   .   .    LP      EQU  100Q       ;TAPE PAST LP HOLE
  729   0080   .   .   .    EW      EQU  200Q       ;TAPE PAST EW HOLE
  730   0000   .   .   .    ;*********************************************
  731   FF62   .   .   .    CNTRLO  EQU  UNITO-1    ;DATA TRANSFER FLAGS: *
  732   0000   .   .   .    ;*********************************************
  733   0001   .   .   .    EOF     EQU    1Q       ;END OF FILE
  734   0002   .   .   .    EVD     EQU    2Q       ;END OF VALID DATA
  735   0004   .   .   .    HRDERR  EQU    4Q       ;HARD ERROR
  736   0008   .   .   .    SFTERR  EQU   10Q       ;SOFT ERROR
  737   0010   .   .   .    HRDER1  EQU   20Q       ;INTERRUPT ERROR FLAG
  738   0020   .   .   .    WRTERR  EQU   40Q       ;WRITE ERROR
  739   0040   .   .   .    DATATR  EQU  100Q       ;DATA RECORDED
  740   0000   .   .   .    ;************************************************
  741   FF61   .   .   .    RELTAK  EQU  CNTRLO-1   ;GAP LENGTH COUNTER
  742   0000   .   .   .    ;********************************************
  743   FF5F   .   .   .    ABSTAK  EQU  RELTAK-2   ;ABSOLUTE TACH COUNTER
  744   0000   .   .   .    ;********************************************
  745   405F   .   .   .    STRTAK  EQU  40137Q     ;STARTING VALUE
  746   0000   .   .   .    ;**************************************************
  747   FF5E   .   .   .    FILNUM  EQU  ABSTAK-1   ;CURRENT FILE NUMBER
  748   FF5D   .   .   .    SFTCNT  EQU  FILNUM-1   ;SOFT ERRORS PER PASS
  749   FF56   .   .   .    OTHER   EQU  SFTCNT-7   ;STORAGE FOR UNIT NOT SEL.
  750   0000   .   .   .    ;*********************************************
  751   FF55   .   .   .    CMND    EQU  OTHER-1    ;CURRENT CTU COMMAND: *
  752   0000   .   .   .    ;*********************************************
  753   0001   .   .   .    RUN     EQU 1Q         ;MOVE TAPE
  754   0002   .   .   .    FWD     EQU 2Q         ;FORWARD
  755   0004   .   .   .    FST     EQU 4Q         ;FAST
  756   0008   .   .   .    REC     EQU 10Q        ;RECORD
  757   0010   .   .   .    USL     EQU  20Q          ;SELECT LEFT UNIT
  758   0020   .   .   .    GEN     EQU 40Q        ;GAP GENERATE
  759   0040   .   .   .    ANR     EQU  100Q         ;LIGHT FOR RIGHT UNIT
  760   0080   .   .   .    ANL     EQU  200Q         ;LIGHT FOR LEFT UNIT
  761   0000   .   .   .    ;*******************************************
  762   0000   .   .   .    ;  INPDEV, OUTDEV, BXSTAT - I/O DEVICES  *
  763   0000   .   .   .    ;*******************************************
  764   0001   .   .   .    LFTCTU  EQU    1Q        ;LEFT CARTRIDGE TAPE UNIT
  765   0002   .   .   .    RGTCTU  EQU    2Q        ;RIGHT CARTRIDGE TAPE UNIT
  766   0004   .   .   .    DISPLY  EQU    4Q        ;DISPLAY
  767   0008   .   .   .    PRINTR  EQU   10Q        ;PRINTER
  768   0010   .   .   .    ALTIO   EQU   20Q        ;ALTERNATE I/O
  769   0020   .   .   .    DATCOM  EQU   40Q        ;DATA COMM
  770   0080   .   .   .    BUFBSY  EQU  200Q        ;BUF HELD BY UNSPECIFIED DEV
  772    FF54    .    .    .   SCNCNT EQU  CMND-1     ;NUM. OF KBSCAN PER CTU SCAN
  773    FF53    .    .    .   CTBLNK EQU  SCNCNT-1   ;BLINK MASK FOR EJECT LIGHTS
  774    FF52    .    .    .   CTBLTM EQU  CTBLNK-1   ;BLINK TIMER
  775    0020    .    .    .   CTBDLY EQU  40Q        ;BLINK DELAY
  776    FF51    .    .    .   HOLCNT EQU  CTBLTM-1   ;HOLE COUNTER
  777    FF50    .    .    .   TPSTAL EQU  HOLCNT-1   ;TAPE STALL COUNTER
  778    0000    .    .    .   ;***************
  779    0000    .    .    .   ; I/O VARIBLES *
  780    0000    .    .    .   ;***************
  781    FF4F    .    .    .   IOCERR EQU  TPSTAL-1   ;I/O ERROR FLAG
  782    0000    .    .    .   ;                        0 = NO ERROR
  783    0000    .    .    .   ;                       -1 = ERROR OCCURRED
  784    FF4E    .    .    .   INPDEV EQU  IOCERR-1   ;CURRENT INPUT DEVICE
  785    FF4D    .    .    .   OUTDEV EQU INPDEV-1 ;CURRENT OUTPUT DEVICE
  786    FF4C    .    .    .   IOCDPT EQU OUTDEV-1 ;DEVICE FLAG POINTER
  787    FF4B    .    .    .   IOSTA3 EQU  IOCDPT-1   ;DEVICE STATUS BYTE 3
  788    FF4A    .    .    .   IOSTA2 EQU  IOSTA3-1   ;DEVICE STATUS BYTE 2
  789    FF49    .    .    .   IOSTA1 EQU  IOSTA2-1   ;DEVICE STATUS BYTE 1
  790    FF48    .    .    .   IOSTA0 EQU  IOSTA1-1   ;DEVICE NUMBER FOR STATUS
  791    FF47    .    .    .   XFRLIM EQU  IOSTA0-1   ;TRANSFER LIMIT
  792    FF46    .    .    .   CMPLIM EQU  XFRLIM-1   ;COMPARE LIMIT
  793    FF3D    .    .    .   B2DBUF EQU  CMPLIM-9   ;BIN TO DECIMAL CONV BUFFER
  794    003D    .    .    .   B2DBFL EQU  B2DBUF-BASE  ;LSB PART OF "B2DBUF"
  795    FF3C    .    .    .   B2DPTR EQU  B2DBUF-1   ;B2DBUF "GET" POINTER (LSB)
  796    FF3B    .    .    .   B2DEND EQU B2DPTR-1 ;B2DBUF END POINTER
  797    0000    .    .    .   ;
  798    0000    .    .    .   ;  I/O CONTROL VARIABLES
  799    0000    .    .    .   ;
  800    FFDB    .    .    .   IOCDEV EQU  PARM1      ;DEVICE FLAG
  801    FFDA    .    .    .   IOCOUT EQU  PARM2      ;OUTPUT DEVICE ACCUMULATOR
  802    FFD9    .    .    .   IOCINP EQU  PARM3      ;INPUT DEVICE ACCUMULATOR
  803    FFD8    .    .    .   IOCTYP EQU  PARM4      ;COMMAND MODIFIER FLAG
  804    FFD7    .    .    .   IOCMND EQU  PARM5      ;COMMAND TYPE FLAG
  805    FFD5    .    .    .   IOCCNT EQU  PARM6      ;DATA COUNT (2 BYTES)
```

```
 807    0000    .    .    .    ;
 808    0000    .    .    .    ;  I/O BUFFER INFORMATION STORAGE
 809    0000    .    .    .    ;
 810    FF3A    .    .    .    B1STAT EQU  B2DEND-1  ;STATUS OF FIRST BUFFER
 811    FF39    .    .    .    B1TYPE EQU  B1STAT-1  ;TYPE (-1=NORM, 0=EOF, 1=EVD
 812    FF38    .    .    .    B1LEN  EQU  B1TYPE-1  ;LENGTH OF RECORD
 813    FF37    .    .    .    B2STAT EQU  B1LEN-1   ;STATUS OF SECOND BUFFER
 814    FF36    .    .    .    B2TYPE EQU  B2STAT-1  ;TYPE (-1=NORM, 0=EOF, 1=EVD
 815    FF35    .    .    .    B2LEN  EQU  B2TYPE-1  ;LENGTH OF RECORD
 816    0000    .    .    .    ;
 817    0000    .    .    .    ;  STORAGE FOR CARTRIDGE TAPE INTERRUPT ROUTINES
 818    0000    .    .    .    ;
 819    FF33    .    .    .    CTIADR EQU  B2LEN-2   ;ADDRESS (HAS SEVERAL USES)
 820    FF31    .    .    .    CTISPT EQU  CTIADR-2  ;POINTER TO BUFFER STATUS
 821    FF2F    .    .    .    CTIBPT EQU  CTISPT-2  ;POINTER TO BUFFER
 822    FF2C    .    .    .    CTICNT EQU  CTIBPT-3  ;GENERAL COUNTERS
 823    FF2B    .    .    .    CTITRL EQU  CTICNT-1  ;RE-READ COUNTER, HOLE CNTR
 824    FF2A    .    .    .    CTICSM EQU  CTITRL-1  ;CHECKSUM COUNTER
 825    FF29    .    .    .    CTISTA EQU  CTICSM-1  ;COMMAND SOURCE FLAG
 826    0000    .    .    .    ;
 827    0000    .    .    .    ;  STORAGE FOR READ AND RECORD
 828    0000    .    .    .    ;
 829    FF27    .    .    .    NXTRED EQU  CTISTA-2  ;PTR INTO BUF FOR NEXT READ
 830    FF25    .    .    .    LSTRED EQU  NXTRED-2  ;PTR INTO BUF FOR READ REPEA
 831    FF24    .    .    .    SWPCTU EQU  LSTRED-1  ;SWAP CTU IN LOGGING MODE
 832    0000    .    .    .    ;                        -1 = SWAP ENABLED
 833    0000    .    .    .    ;                         0 = DISABLED
 834    FF23    .    .    .    SAVINP EQU  SWPCTU-1  ;"INPDEV" SAVE FOR LOCAL RCR
 835    FF22    .    .    .    SAVOUT EQU  SAVINP-1  ;SAVE OUTDEV DURING LCL READ
 836    0000    .    .    .    ;
 837    0000    .    .    .    ;  DATA FOR FORMAT DISPLAY STORAGE
 838    0000    .    .    .    ;
 839    FF21    .    .    .    ENDCOL EQU  SAVOUT-1  ;ENDING COLUMN AND ROW FOR
 840    FF20    .    .    .    ENDROW EQU  ENDCOL-1     ;PREV NON-PROTECTED FIELD
 842   0000    .   .   .   ;
 843   0000    .   .   .   ;  EXTENDED MAIN CODE RAM AREA
 844   0000    .   .   .   ;
 845   FE80    .   .   .   XTRASP EQU  177200Q
 846   0000    .   .   .   ;*****************************************
 847   FE7F    .   .   .   DEVFLG EQU  XTRASP-1  ;DEVICE PRESENT FLAG
 848   0000    .   .   .   ;*****************************************
 849   0080    .   .   .   CTUIN  EQU  2000      ;CTU CODE PRESENT
 850   0040    .   .   .   ALTIN  EQU  1000      ;ALTERNATE I/O PRESENT
 851   0000    .   .   .   ;********************
 852   0000    .   .   .   ; PRINTER VARIABLES *
 853   0000    .   .   .   ;********************
 854   FE7D    .   .   .   PTRBBG EQU  DEVFLG-2  ;START OF PRINTER BUFFER
 855   FE7B    .   .   .   PTRSPT EQU  PTRBBG-2  ;LOAD POINTER
 856   FE79    .   .   .   PTRBPT EQU  PTRSPT-2  ;UNLOAD POINTER
 857   FE78    .   .   .   PTRABT EQU  PTRBPT-1  ;PRINTER ERROR FLAG
 858   0000    .   .   .   ;                        =  0, NO PRINTER ERROR
 859   0000    .   .   .   ;                        = -1, PRINT ERROR OCCURRED
 860   FE77    .   .   .   PTRFLG EQU  PTRABT-1  ;PRINTER TYPE FLAG
 861   0000    .   .   .   ;                        = 0, NO PRINTER
 862   0000    .   .   .   ;                        = 1, PARALLEL INTERFACE
 863   0000    .   .   .   ;                        = 2, RS-232 INTERFACE
 865   0000   .   .   .   ;*********************************
 866   0000   .   .   .   ; ENTRY VECTORS TO I/O ROUTINES *
 867   0000   .   .   .   ;*********************************
 868   0000   .   .   .   ;
 869   0000   .   .   .   ;  KEYBOARD INITIATED FUNCTIONS
 870   0000   .   .   .   ;
 871   2800   .   .   .   IOORG  EQU  24000Q     ;START OF I/O CODE
 872   2802   .   .   .   IOCKEY EQU  IOORG+2    ;I/O CONTROL KEY
 873   2805   .   .   .   REDKEY EQU  IOCKEY+3   ;READ KEY
 874   2808   .   .   .   CTLRED EQU  REDKEY+3   ;CONTROL READ KEY
 875   280B   .   .   .   RECKEY EQU  CTLRED+3   ;RECORD KEY
 876   280E   .   .   .   SELKEY EQU  RECKEY+3   ;SELECT KEY
 877   2811   .   .   .   TSTCTU EQU  SELKEY+3   ;CTU SELF-TEST
 878   2814   .   .   .   CONDTN EQU  TSTCTU+3   ;CONDITION CARTRIDGE TAPES
 879   2817   .   .   .   RSTCTU EQU  CONDTN+3   ;SOFT RESET FOR CTU
 880   0000   .   .   .   ;
 881   0000   .   .   .   ;  EXTERNALLY INITIATED FUNCTIONS
 882   0000   .   .   .   ;
 883   281A   .   .   .   IOCNTL EQU  RSTCTU+3   ;I/O CONTROL ESCAPE SEQUENCE
 884   281D   .   .   .   IOSTGO EQU  IOCNTL+3   ;SEND DEVICE STATUS
 885   2820   .   .   .   IODNGO EQU  IOSTGO+3   ;SEND COMPLETION CODE
 886   2823   .   .   .   IORDGO EQU  IODNGO+3   ;SEND I/O RECORD
 887   2826   .   .   .   RCRDGO EQU  IORDGO+3   ;START REMOTE RECORD FUNCTIO
 888   2829   .   .   .   BNRYGO EQU  RCRDGO+3   ;SEND BINARY DATA
 889   282C   .   .   .   CTDCDP EQU  BNRYGO+3   ;SEND BINARY FILE
 890   0000   .   .   .   ;*********************
 891   0000   .   .   .   ; INTERNAL ROUTINES *
 892   0000   .   .   .   ;*********************
 893   282F   .   .   .   CTMON  EQU  CTDCDP+3   ;MONITOR CARTRIDGE DRIVES
 894   2832   .   .   .   PTTPLN EQU  CTMON+3    ;PUT TOP LINE ONTO I/O DEV'S
 895   2835   .   .   .   DOOCTI EQU  PTTPLN+3   ;INITIAL CTU INTERRUPT VECTO
 896   2837   .   .   .   RDABRT EQU  DOOCTI+2   ;ABORT USER INITIATED READ
 897   283A   .   .   .   BSYCHK EQU  RDABRT+3   ;WAIT UNTIL TAPE I/O DONE
 898   283D   .   .   .   CTINTR EQU  BSYCHK+3   ;CTU INTERRUPT ROUTINE
 900   0000   .   .   .   ;********************
 901   0000   .   .   .   ; TERMINAL START-UP *
 902   0000   .   .   .   ;********************
 903   0000   .   .   .          ORG  0Q
 904   0000   .   .   .   BEGIN  EQU  $
 905   0000   50  .   .          DB   VERSN      ;ROM PRESENT FLAGS
 906   0001   00  .   .          DB   BEGIN/256   ;(= MOV D,B; NOP)
 907   0002   F3  .   .          DI              ;DISABLE INTERRUPTS
 908   0003   3E  83  .          MVI  A,SETROM+TMIEN+TMRON
 909   0005   C3  B4  00         JMP  GO         ;GO TO START UP ROUTINE
 910   0008   .   .   .   ;*****************************
 911   0008   .   .   .   ; FIRMWARE INVOKED INTERRUPT *
 912   0008   .   .   .   ;*****************************
 913   0008   E9  .   .          PCHL            ;USE AS PCHL SUBROUTINE CALL
 914   0009   .   .   .          ORG  BEGIN+20Q
 915   0010   .   .   .   ;**************************
 916   0010   .   .   .   ; TOP PLANE INTERRUPT 20B *
 917   0010   .   .   .   ;**************************
 918   0010   F5  .   .          PUSH PSW        ;SAVE A-REGISTER AND FLAGS
 919   0011   B7  .   .          ORA  A          ;CLEAR C-FLAG
 920   0012   3E  32  .          MVI  A,TWO      ;SET INTERRUPT CODE
 921   0014   C3  EB  15         JMP  INTRPT     ;HANDLE UNKNOWN INTERRUPTS
 922   0017   .   .   .          ORG  BEGIN+30Q
 923   0018   .   .   .   ;******************
 924   0018   .   .   .   ; TIMER INTERRUPT *
 925   0018   .   .   .   ;******************
 926   0018   F5  .   .          PUSH PSW        ;SAVE A-REGISTER, FLAGS
 927   0019   C5  .   .          PUSH B            ;AND REGISTER B AND C
 928   001A   3E  33  .          MVI  A,THREE    ;SET INTERRUPT CODE
 929   001C   C3  A4  07         JMP  TMINTR     ;CONTINUE TIMER ROUTINE
 930   001F   .   .   .          ORG  BEGIN+40Q
 931   0020   .   .   .   ;**********************
 932   0020   .   .   .   ; DATA COMM INTERRUPT *
 933   0020   .   .   .   ;**********************
 934   0020   F5  .   .          PUSH PSW        ;SAVE A-REGISTER AND FLAGS
 935   0021   3E  34  .          MVI  A,FOUR     ;SET INTERRUPT CODE
 936   0023   C3  34  12         JMP  DCMINT     ;CONTINUE INTERRUPT PROCESS
 937   0026   .   .   .          ORG  BEGIN+50Q
 938   0028   .   .   .   ;***********************
 939   0028   .   .   .   ; I/O DEVICE INTERRUPT *
 940   0028   .   .   .   ;***********************
 941   0028   F5  .   .          PUSH PSW        ;SAVE A-REG, STATUS
 942   0029   E5  .   .          PUSH H            ;AND H,L
 943   002A   3E  35  .          MVI  A,FIVE     ;SET INTERRUPT CODE
 944   002C   C3  AD  15         JMP  IOINTR     ;CONTINUE I/O ROUTINE
 945   002F   .   .   .          ORG  BEGIN+60Q
```

```
  947   0030    .   .   .   ;**************************
  948   0030    .   .   .   ; TOP PLANE INTERRUPT 60B *
  949   0030    .   .   .   ;**************************
  950   0030    F5  .   .          PUSH PSW        ;SAVE A-REGISTER AND FLAGS
  951   0031    B7  .   .          ORA  A          ;CLEAR THE C-FLAG
  952   0032    3E  36  .          MVI  A,SIX      ;SET INTERRUPT CODE
  953   0034    C3  EB  15         JMP  INTRPT     ;HANDLE UNKNOWN INTERRUPTS
  954   0037    .   .   .          ORG  BEGIN+70Q
  955   0038    .   .   .   ;***********************
  956   0038    .   .   .   ; TEST POINT INTERRUPT *
  957   0038    .   .   .   ;***********************
  958   0038    F5  .   .          PUSH PSW        ;SAVE A-REGISTER AND FLAGS
  959   0039    B7  .   .          ORA  A          ;CLEAR THE C-FLAG
  960   003A    3E  37  .          MVI  A,SEVEN    ;SET INTERRUPT CODE
  961   003C    C3  EB  15         JMP  INTRPT     ;HANDLE UNKNOWN INTERRUPTS
  962   003F    .   .   .          ORG  BEGIN+100Q
  964   0040    .    .    .   ;********************************
  965   0040    .    .    .   ; VECTORS TO MAIN CODE ROUTINES *
  966   0040    .    .    .   ;********************************
  967   0040   C3   DA   1C   ZDSPMS:JMP   DSPMSG    ;DISPLAY MESSAGE
  968   0043   C3   0E   1D          JMP   RSTDSP    ;RESTORE NORMAL DISPLAY
  969   0046   C3   62   12          JMP   DCNUM     ;ACCUMULATE DIGIT AND SIGN
  970   0049   C3   86   12          JMP   DCPLUS      ;FOR PARAMETERIZED ESCAPE
  971   004C   C3   8B   12          JMP   DCMNUS      ;SEQUENCES
  972   004F   C3   95   04          JMP   ESCEND    ;TERMINATE ESCAPE SEQUENCE
  973   0052   C3   17   10          JMP   CHKLIM
  974   0055   C3   70   10          JMP   CLBLXF
  975   0058   C3   CA   16          JMP   SBLXF0
  976   005B   C3   CD   16          JMP   SBLXFA    ;KEYBOARD INITIATED BLK XFR
  977   005E   C3   63   17          JMP   STRTBL    ;START BLOCK RECORD
  978   0061   C3   27   1D          JMP   CURPH     ;HOME CURSOR (-XMIT ONLY)
  979   0064   C3   07   11          JMP   CURPHD    ;CURSOR HOME DOWN
  980   0067   C3   0A   15          JMP   FRECNT    ;CHECK NUMBER OF FREE BLOCKS
  981   006A   C3   13   06          JMP   PTBLK     ;RELEASE BLOCKS FROM DISPLAY
  982   006D   C3   3C   1C          JMP   CLEARL    ;CLEAR LINE
  983   0070   C3   8F   10          JMP   CLEARS    ;CLEAR DISPLAY FROM CURSOR
  984   0073   C3   FA   14          JMP   FNDTB2    ;SET BIT N (B-REG = N)
  985   0076   C3   1D   12          JMP   SDTERM    ;SEND TERMINATORS
  986   0079   C3   F6   16          JMP   SDTRM1    ;SEND TERMINATOR ONLY
  987   007C   C3   C1   17          JMP   XPUTDC    ;TRANSMIT CHARACTER IN A-REG
  988   007F   C3   8C   0D          JMP   TRMTST    ;TERMINAL SELF-TEST
  989   0082   C3   30   03          JMP   CHINIO    ;EXECUTE CHARACTER FUNCTION
  990   0085   C3   93   25          JMP   INITDO    ;INIT FOR DISPLAY GET
  991   0088   C3   2C   24          JMP   GETDSP    ;GET DISPLAY BYTE
  992   008B   C3   6F   0A          JMP   LNFEED    ;DO LINE FEED
  993   008E   C3   58   23          JMP   EXPAND    ;EXPAND DISPLAY CONTROL CHAR
  994   0091   C3   87   0B          JMP   NXTCHR    ;GET NEXT DISPLAY CHARACTER
  995   0094   C3   FC   04          JMP   GETDCM    ;PROCESS DATA COMM INPUT
  996   0097   C3   E4   0A          JMP   MLKSC0    ;LOCATE FIRST UNLOCKED ROW
  997   009A   C3   B9   0A          JMP   MLKOF0    ;TURN OFF MEMORY LOCK
  998   009D   C3   54   12          JMP   HANGU0    ;HANG TERMINAL ON FATAL ERRO
  999   00A0   27   0F   .           DW    BUFMSG    ;BUFFER OVERFLOW MESSAGE
 1000   00A2   C3   99   12          JMP   DCTEST    ;DATA COMM SELF-TEST
 1001   00A5   C3   93   15          JMP   IORMG0    ;EXECUTE CODE FROM OPTION RO
 1002   00A8   C3   2E   08          JMP   BN2DEC    ;CONVERT BINARY TO DECIMAL
 1003   00AB   C3   1D   08          JMP   BN2DE0    ;CONVERT SINGLE BYTE TO DEC
 1004   00AE   C3   A4   06          JMP   RCADRA    ;LOCATE CURSOR LOCATION
 1005   00B1   C3   59   10          JMP   GTMODE    ;CHECK FOR PAGE MODE
```

```
=======================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                   PAGE  28
=======================================================================
 1007    00B4    .    .    .    ;**************************************
 1008    00B4    .    .    .    ; TERMINAL RESET - START UP TERMINAL *
 1009    00B4    .    .    .    ;**************************************
 1010    00B4    .    .    .    GO      EQU   $
 1011    00B4    D3   70   .            OUT   PROCSR      ;SET INITIAL PROCESSOR STATE
 1012    00B6    32   F5   FF           STA   PRCCTL      ;SET PROCESSOR STATE
 1013    00B9    31   60   91           LXI   SP,STACK    ;SET STACK POINTER
 1014    00BC    3A   CD   FF           LDA   ECONTF
 1015    00BF    FE   C3   .            CPI   JMP         ;FIRST TURN ON?
 1016    00C1    .    .    .    ;************************************************
 1017    00C1    C2   F4   00           JNZ   INIT        ;YES - INITIALIZE TERMINAL
 1018    00C4    .    .    .    ;************************************************
 1019    00C4    3A   F8   FF           LDA   CMFLGS      ;NO - GET COMMON FLAGS
 1020    00C7    E6   04   .            ANI   FRCRST      ;FORCE FULL RESET?
 1021    00C9    C2   F4   00           JNZ   INIT        ;YES - INITIALIZE TERMINAL
 1022    00CC    21   D0   FF           LXI   H,RSTTMR    ;NO - GET SOFT RESET TIMER
 1023    00CF    B6   .    .            ORA   M           ;FULL RESET ACTIVE?
 1024    00D0    CA   D8   00           JZ    GO010       ;NO - START SOFT RESET
 1025    00D3    FE   32   .            CPI   SFTDLY      ;STILL IN SOFT RESET START?
 1026    00D5    .    .    .    ;                           (CAUSED BY CONTACT BOUNCE)
 1027    00D5    C2   F4   00           JNZ   INIT        ;NO - DO FULL RESET
 1028    00D8    .    .    .    GO010   EQU   $           ;YES - RESTART SOFT RESET
 1029    00D8    36   32   .            MVI   M,SFTDLY    ;NO - SET 0.5 SEC TIME OUT
 1030    00DA    .    .    .    ;****************
 1031    00DA    .    .    .    ; DO SOFT RESET *
 1032    00DA    .    .    .    ;****************
 1033    00DA    .    .    .    GO1     EQU   $           ;ENTRY FOR SOFT RESET
 1034    00DA    32   6E   FF           STA   DFLGS       ;CLEAR DATA TRANSFER FLAGS
 1035    00DD    3E   07   .            MVI   A,RSETKB
 1036    00DF    CD   08   48           CALL  ZKBCTL      ;RESET THE KEYBOARD
 1037    00E2    3E   02   .            MVI   A,RSETDC
 1038    00E4    CD   4D   12           CALL  DCMCT1      ;RESET THE DATA COMM
 1039    00E7    21   17   28           LXI   H,RSTCTU    ;RESET CARTRDIGE TAPES
 1040    00EA    CD   93   15           CALL  IORMGO        ;IF CTU CODE PRESENT
 1041    00ED    CD   0E   1D           CALL  RSTDSP      ;RESTORE NORMAL DISPLAY
 1042    00F0    FB   .    .            EI                ;ENABLE INTERRUPTS
 1043    00F1    C3   20   02           JMP   START       ;RESTART THE WAIT LOOP
```

```
=====================================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                                PAGE  29
=====================================================================================
1045    00F4   .    .    .   ;*********************************************
1046    00F4   .    .    .   ; INIT - DO COMPLETE TERMINAL INITIALIZATION *
1047    00F4   .    .    .   ;*********************************************
1048    00F4   .    .    .   INIT    EQU  $
1049    00F4   AF   .    .           XRA  A          ;CLEAR TO ZERO
1050    00F5   32   CD   FF          STA  ECUNTF     ;CLEAR "JMP" TO FORCE FULL
1051    00F8   .    .    .   ;                         RESET
1052    00F8   21   00   91          LXI  H,FSTRAM   ;SET FIRST ADDRESS
1053    00FB   .    .    .   INI010  EQU  $
1054    00FB   77   .    .           MOV  M,A        ;SET BYTE TO ZERO
1055    00FC   2C   .    .           INR  L          ;ALL BYTES DONE?
1056    00FD   C2   FB   00          JNZ  INI010     ;NO - CLEAR NEXT BYTE
1057    0100   .    .    .   ;
1058    0100   .    .    .   ;  CLEAR SLOW RAM AREA
1059    0100   .    .    .   ;
1060    0100   5D   .    .           MOV  E,L        ;SET E = 0 FOR 256 BYTES
1061    0101   26   FC   .           MVI  H,IOBUF1/256  ;SET START ADDRESS
1062    0103   .    .    .   INI020  EQU  $
1063    0103   CD   FE   10          CALL CLRAL1     ;CLEAR A 256 BYTE SECTION
1064    0106   BC   .    .           CMP  H          ;ALL SECTIONS CLEARED?
1065    0107   C2   03   01          JNZ  INI020     ;NO - CONTINUE CLEARING
1066    010A   .    .    .   ;***************************
1067    010A   .    .    .   ; LOCATE NON-DISPLAY SPACE *
1068    010A   .    .    .   ;***************************
1069    010A   21   FF   CF          LXI  H,BFSPCE   ;SET UPPER BOUNDARY ADDRESS
1070    010D   22   8B   FF          SHLD BUFEND       ;OF NON-DISPLAY BUFFER ARE
1071    0110   06   B0   .           MVI  B,LWBUF    ;SET B TO MSB OF LOWER LIMIT
1072    0112   CD   B0   04          CALL FNDRAM
1073    0115   22   8D   FF          SHLD BUFBGN     ;STORE BUFFER START ADDRESS
1074    0118   .    .    .   ;
1075    0118   .    .    .   ;  LOCATE DISPLAY SPACE
1076    0118   .    .    .   ;
1077    0118   21   FF   FB          LXI  H,DSPLIM   ;SET UPPER BOUNDARY ADDRESS
1078    011B   22   A8   FF          SHLD DSPEND       ;OF DISPLAY AREA
1079    011E   06   D0   .           MVI  B,LWDSP    ;SET B TO MSB OF LOWER LIMIT
1080    0120   CD   B0   04          CALL FNDRAM
1081    0123   22   AA   FF          SHLD DSPBGN     ;STORE DISPLAY START ADDRESS
```

```
======================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
======================================================================
1083    0126   .    .    .   ;*********************************************
1084    0126   .    .    .   ; INITIALIZE PROCESSOR BOARD STATE, KEYBOARD, *
1085    0126   .    .    .   ; AND DATA COMM                               *
1086    0126   .    .    .   ;*********************************************
1087    0126   3E   83   .           MVI  A,SETROM+TMIEN+TMRON
1088    0128   32   F5   FF          STA  PRCCTL     ;ENABLE ROM'S AND TIMER
1089    012B   3E   C9   .           MVI  A,RET      ;PUT RETURN CODE INTO
1090    012D   32   65   91          STA  INTVEC        ;INTERRUPT VECTOR AND
1091    0130   32   68   91          STA  SCNVEC        ;DISPLAY SCAN VECTOR
1092    0133   .    .    .   ;                 **********************************
1093    0133   .    .    .   ;                 * INTERRUPTS ARE ENABLED BY THE *
1094    0133   .    .    .   ;                 * DISPLAY ROUTINES USED DURING  *
1095    0133   .    .    .   ;                 * INITIALIZATION OF SOFT KEYS   *
1096    0133   .    .    .   ;                 **********************************
1097    0133   CD   02   48          CALL ZINIKB     ;SET JUMPERS AND DC SWITCHES
1098    0136   CD   08   50          CALL ZINIDC     ;FETCH BUFFER REQUIREMENTS
1099    0139   CD   CB   04          CALL GETBUF     ;ALLOCATE BUFFER SPACE
1100    013C   CD   0B   50          CALL ZIN2DC     ;COMPLETE DATA COMM INIT
1101    013F   DA   54   12          JC   HANGUO        ;(PROCESS ERROR IF ANY)
1102    0142   .    .    .   ;**********************************
1103    0142   .    .    .   ; SET DEFAULT I/O CONFIGURATION *
1104    0142   .    .    .   ;**********************************
1105    0142   21   02   01          LXI  H,1*256+2  ;OUTPUT = RIGHT CTU (2)
1106    0145   22   4D   FF          SHLD OUTDEV      ;INPUT = LEFT CTU (1)
1107    0148   2A   35   28          LHLD DOOCTI     ;SET INITIAL CARTRIDGE TAPE
1108    014B   22   E1   FF          SHLD CTIVEC        ;INTERRUPT VECTOR
1109    014E   3E   C3   .           MVI  A,JMP      ;SET JUMP COMMAND FOR
1110    0150   32   E0   FF          STA  CTIJMP        ;CTU INTERRUPT VECTOR
1111    0153   .    .    .   ;*******************************************
1112    0153   .    .    .   ; IDENTIFY OPTION I/O INCLUDED IN TERMINAL *
1113    0153   .    .    .   ;*******************************************
1114    0153   21   02   60          LXI  H,ZINIAL   ;INITIALIZE ALTERNATE I/O
1115    0156   CD   93   15          CALL IORMGO        ;DEVICE
1116    0159   3E   00   .           MVI  A,0           ;(SET FOR NO ALTERNATE I/O
1117    015B   DA   66   01          JC   INI110     ;BYPASS INIT IF NO ALT I/O
1118    015E   CD   CB   04          CALL GETBUF        ;ELSE, ALLOCATED BUFFER
1119    0161   CD   05   60          CALL ZIN2AL        ;AND CONTINUE INIT
1120    0164   3E   40   .           MVI  A,ALTIN    ;SET ALT I/O PRESENT BIT
1121    0166   .    .    .    INI110 EQU  $
1122    0166   47   .    .           MOV  B,A        ;SAVE ALTERNATE I/O STATUS
1123    0167   21   00   28          LXI  H,IOORG    ;SET I/O START ADDRESS
1124    016A   CD   A3   15          CALL IORMG1     ;DOES I/O CODE EXIST?
1125    016D   78   .    .           MOV  A,B          ;(GET CURRENT I/O OPTIONS)
1126    016E   C2   77   01          JNZ  INI130     ;NO - DON'T SET I/O BIT
1127    0171   F6   80   .           ORI  CTUIN      ;ELSE SET CTU PRESENT BIT
1128    0173   21   FD   FF          LXI  H,TRMTYP   ;SET TERM TYPE TO INDICATE
1129    0176   34   .    .           INR  M             ;I/O CODE INCLUDED
1130    0177   .    .    .    INI130 EQU  $
1131    0177   32   7F   FE          STA  DEVFLG     ;SET I/O OPTIONS FLAG
```

```
 1133   017A   .    .    .    ;****************************************
 1134   017A   .    .    .    ; GENERATE FREE BLOCKS LIST FOR DISPLAY *
 1135   017A   .    .    .    ;****************************************
 1136   017A   2A   A8   FF           LHLD DSPEND      ;COMPUTE ADDRESS OF HIGHEST
 1137   017D   11   F1   FF           LXI  D,1-BLKSZ   ;ADDRESSED DISPLAY BLOCK
 1138   0180   19   .    .            DAD  D
 1139   0181   7D   .    .            MOV  A,L         ;COMPUTE ADDRESS OF LSB PART
 1140   0182   F6   0F   .            ORI  BLKSM         ;OF PREVIOUS LINE POINTER
 1141   0184   6F   .    .            MOV  L,A           ;IN HIGHEST ADDRESSED
 1142   0185   2B   .    .            DCX  H             ;DISPLAY BLOCK
 1143   0186   36   00   .            MVI  M,0         ;SET IT TO ZERO TO INDICATE
 1144   0188   2B   .    .            DCX  H             ;END OF FREE LIST
 1145   0189   EB   .    .            XCHG             ;SET NEXT BLOCK LINK OF
 1146   018A   2A   AA   FF           LHLD DSPBGN        ;LOWEST ADDRESSED DISPLAY
 1147   018D   73   .    .            MOV  M,E           ;BLOCK TO POINT TO MSB
 1148   018E   23   .    .            INX  H             ;PART OF NEXT LINE LINK IN
 1149   018F   72   .    .            MOV  M,D           ;HIGHEST BLOCK
 1150   0190   EB   .    .            XCHG             ;SWAP HIGH AND LOW ADDRESSES
 1151   0191   13   .    .            INX  D           ;ADJUST LOW ADDR TO LOW LIMI
 1152   0192   .    .    .    ;                          FOR LINKING DISPLAY BLOCKS
 1153   0192   2B   .    .            DCX  H           ;SET FREE BLOCKS HEAD TO LSB
 1154   0193   22   AC   FF           SHLD FRBLKS        ;PART OF NEXT LINE POINTER
 1155   0196   .    .    .    ;                          IN HIGHEST BLOCK
 1156   0196   D6   0E   .            SUI  BLKSZ-2     ;SET B,L TO ADDRESS OF MSB
 1157   0198   44   .    .            MOV  B,H           ;PART OF NEXT BLOCK POINTE
 1158   0199   6F   .    .            MOV  L,A           ;IN HIGHEST DISPLAY BLOCK
```

```
1160   019A   .   .   .   ;********************
1161   019A   .   .   .   ; CHAIN FREE BLOCKS *
1162   019A   .   .   .   ;********************
1163   019A   .   .   .   ;
1164   019A   .   .   .   ;  B,A = ADDRESS OF UPPER BYTE IN NEXT LOWER BLOCK
1165   019A   .   .   .   ;  D,E = LOWER LIMIT OF DISPLAY AREA
1166   019A   .   .   .   ;  H,L = ADDRESS OF MSB PART OF NEXT BLOCK LINK
1167   019A   .   .   .   ;     IN CURRENT BLOCK
1168   019A   .   .   .   ;
1169   019A   .   .   .   INI210 EQU  $
1170   019A   7D  .   .          MOV  A,L        ;COMPUTE ADDRESS OF UPPERMOS
1171   019B   D6  02  .          SUI  2            ;BYTE IN NEXT LOWER BLOCK
1172   019D   D2  A1  01         JNC  INI220
1173   01A0   05  .   .          DCR  B
1174   01A1   .   .   .   INI220 EQU  $
1175   01A1   70  .   .          MOV  M,B        ;LINK CURRENT BLOCK TO NEXT
1176   01A2   2B  .   .          DCX  H            ;LOWER BLOCK
1177   01A3   77  .   .          MOV  M,A
1178   01A4   D6  0E  .          SUI  BLKSZ-2    ;SET H,L TO ADDRESS OF MSB
1179   01A6   6F  .   .          MOV  L,A          ;PART OF NEXT BLOCK LINK I
1180   01A7   60  .   .          MOV  H,B          ;NEXT LOWER BLOCK
1181   01A8   93  .   .          SUB  E          ;COMPARE AGAINST LOWER LIMIT
1182   01A9   78  .   .          MOV  A,B
1183   01AA   9A  .   .          SBB  D          ;DISPLAY AREA EXHAUSTED?
1184   01AB   D2  9A  01         JNC  INI210     ;NO - CONTINUE LINKING BLOCK
1185   01AE   .   .   .   ;                      YES - SET UP INITIAL DISPLAY
```

```
=====================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                                PAGE  33
=====================================================================================
 1187   01AE    .    .    .    ;
 1188   01AE    .    .    .    ;  SET UP INITIAL SOFT KEYS DISPLAY
 1189   01AE    .    .    .    ;
 1190   01AE   CD   CB   05              CALL INITDS     ;START A NEW DISPLAY LIST
 1191   01B1   2B    .    .              DCX  H          ;SET SOFT KEY START ADDRESS
 1192   01B2   22   A6   FF              SHLD SFTKYS       ;TO FIRST CHARACTER
 1193   01B5   3E   80    .              MVI  A,CRTOFF   ;SET CURRENT AND LAST ROW
 1194   01B7   32   C0   FF              STA  CURROW       ;TO CONTROL FOR DISPLAY OF
 1195   01BA   32   C7   FF              STA  LSTROW
 1196   01BD    .    .    .    ;
 1197   01BD    .    .    .    ;  SET UP KEY DEFINITIONS
 1198   01BD    .    .    .    ;
 1199   01BD   01   4E   FE              LXI  B,DSPSTR-1
 1200   01C0   21   DA   14              LXI  H,ATBLIN   ;TRANSFER ATTRIBUTE LINE
 1201   01C3   CD   20   0B              CALL MOVCHR
 1202   01C6   0E   08    .              MVI  C,NMFCTK   ;SET NUMBER OF KEYS TO DEFIN
 1203   01C8    .    .    .    ;
 1204   01C8    .    .    .    ;  BUILD ATTRIBUTE LINE
 1205   01C8    .    .    .    ;
 1206   01C8    .    .    .    INI310 EQU  $
 1207   01C8   3E   39    .              MVI  A,ZERO+9   ;COMPUTE FUNCTION KEY NUMBER
 1208   01CA   91    .    .              SUB  C
 1209   01CB   32   43   FE              STA  DSPSTR-ATBLEN+2
 1210   01CE    .    .    .    ;
 1211   01CE    .    .    .    ;  BUILD DEFINITION LINE
 1212   01CE    .    .    .    ;
 1213   01CE   3E   78    .              MVI  A,SMALLX   ;COMPUTE CHAR AFTER <ESC>
 1214   01D0   91    .    .              SUB  C              ;(LOWER CASE <P>-<W>)
 1215   01D1   21   4D   FE              LXI  H,DSPSTR-CHRLOC
 1216   01D4   77    .    .              MOV  M,A        ;SET DATA CHARACTER
 1217   01D5   2E   41    .              MVI  L,DSPSTR-ATBLEN-BASE2
 1218   01D7   C5    .    .              PUSH B          ;TRANSFER SOFT KEY DEFINITIO
 1219   01D8   CD   CD   0F              CALL XMS2DS        ;TO DISPLAY MEMORY
 1220   01DB   C1    .    .              POP  B
 1221   01DC   0D    .    .              DCR  C          ;ALL KEYS DEFINED?
 1222   01DD   C2   C8   01              JNZ  INI310     ;NO - DO NEXT KEY
```

```
01E0   .   .   .   ;*********************************************
01E0   .   .   .   ; SOFT KEYS DONE - SET INITIAL DISPLAY STATE *
01E0   .   .   .   ;*********************************************
01E0   AF  .   .          XRA  A          ;CLEAR LAST LINE POINTER
01E1   32  A1  FF         STA  LLINE
01E4   3D  .   .          DCR  A          ;SET DISPLAY TYPE TO SOFT
01E5   32  AE  FF         STA  DSPTYP       ;KEY DISPLAY
01E8   CD  27  1D         CALL CURPH      ;HOME THE CURSOR
01EB   CD  69  21         CALL SWAP       ;SAVE SOFT KEY PARAMETERS
01EE   .   .   .   ;***********************************
01EE   .   .   .   ; INITIALIZE FIRST LINE OF DISPLAY *
01EE   .   .   .   ;***********************************
01EE   CD  CB  05         CALL INITDS     ;START A NEW DISPLAY LIST
01F1   .   .   .   ;*************************
01F1   .   .   .   ; INITIALIZE I/O DEVICES *
01F1   .   .   .   ;*************************
01F1   .   .   .   ;
01F1   .   .   .   ;      PRINTER INITIALIZATION ROUTINE
01F1   .   .   .   ;
01F1   .   .   .   ;  CHECK FOR 9866 PRINTER FIRST
01F1   .   .   .   ;
01F1   3A  00  8D         LDA  PTRST1     ;GET STATUS FROM 9866 PCA
01F4   B7  .   .          ORA  A          ;IS INTERFACE INSTALLED?
01F5   CA  00  02         JZ   PTRI10     ;NO - LOOK FOR RS-232 PRNTR
01F8   3A  02  8D         LDA  PTRCL1     ;YES - CLEAR THE PRINTER
01FB   3E  01  .          MVI  A,1        ;SET PRINTER FLAG FOR
01FD   C3  12  02         JMP  PTR120       ;9866 PRINTER (= 1)
0200   .   .   .   ;
0200   .   .   .   ;     RS-232         PRINTER 2
0200   .   .   .   ;
0200   .   .   .   PTRI10 EQU $
0200   3A  20  85         LDA  PTRST2     ;GET STATUS FROM RS-232 PCA
0203   B7  .   .          ORA  A          ;IS RS-232 PCA INSTALLED?
0204   CA  12  02         JZ   PTR120     ;NO - SET FOR NO PRINTER
0207   .   .   .   ;
0207   3A  40  85         LDA  PTRCF2     ;YES - GET CONFIG. STRAPS
020A   E6  1F  .          ANI  PTRBD2     ;ISOLATE BAUD AND PARITY
020C   17  .   .          RAL             ;ADJUST FOR CONTROL OUTPUT
020D   32  40  85         STA  PTROT2     ;SET BOARD TO CONFIGURATION
0210   3E  02  .          MVI  A,2        ;SET FLAG FOR RS-232 PRINTER
0212   .   .   .   ;
0212   .   .   .   PTR120 EQU  $
0212   32  77  FE         STA  PTRFLG     ;SET PRINTER FLAG
0215   21  6A  0F         LXI  H,TRMRDY   ;DISPLAY "TERMINAL READY"
0218   CD  D7  1C         CALL DSPMS1       ;MESSAGE
021B   3E  C3  .          MVI  A,JMP      ;SET JUMP COMMAND FOR
021D   32  CD  FF         STA  ECUNIF       ;CHARACTER FUNCTION VECTOR
```

```
1272   0220    .    .    .    ;*****************************************
1273   0220    .    .    .    ; INITIALIZE FLAGS AND RANGE TABLE ADDRESS *
1274   0220    .    .    .    ;*****************************************
1275   0220    .    .    .    START    EQU  $
1276   0220    CD   95   04            CALL ESCEND      ;RESET NORMAL RANGE TABLE
1277   0223    3A   29   48            LDA  FRSALT      ;SET INITIAL ALTERNATE
1278   0226    32   72   FF            STA  CHRSET        ;CHARACTER SET
1279   0229    CD   23   20            CALL CRADV1      ;CLEAR CURSOR ADVANCE FLAG
1280   022C    3D   .    .             DCR  A           ;CLEAR SPOW LATCH
1281   022D    32   6C   FF            STA  SPOWL
```

```
 1283   0230   .    .    .   ;
 1284   0230   .    .    .   ;  WAIT LOOP
 1285   0230   .    .    .   ;
 1286   0230   .    .    .   WTLOOP EQU  $
 1287   0230   31   60   91         LXI  SP,STACK  ;SET STACK POINTER
 1288   0233   CD   30   05         CALL GETDC1    ;SET DISPLAY CURSOR
 1289   0236   .    .    .   ;****************************
 1290   0236   .    .    .   ; CHECK FOR DATA COMM INPUT *
 1291   0236   .    .    .   ;****************************
 1292   0236   .    .    .   WTL010 EQU  $
 1293   0236   CD   FC   04         CALL GETDCM    ;GET DATA COMM INPUT IF ANY
 1294   0239   .    .    .   ;***************************
 1295   0239   .    .    .   ; CHECK FOR KEYBOARD INPUT *
 1296   0239   .    .    .   ;***************************
 1297   0239   3A   6F   FF         LDA  MFLGS2    ;GET MODE FLAGS
 1298   023C   E6   08   .          ANI  ESCINP    ;ESCAPE SEQUENCE LOCK OUT?
 1299   023E   C2   5F   02         JNZ  WTL020    ;YES - IGNORE KEYBOARD
 1300   0241   3A   6E   FF         LDA  DFLGS     ;NO - GET DATA TRANSFER FLAG
 1301   0244   E6   10   .          ANI  FCTK2D    ;FUNCTION KEY TO DISPLAY?
 1302   0246   C4   42   15         CNZ  GTFCTK    ;YES - GET FUNCTION KEY CHAR
 1303   0249   C2   74   02         JNZ  WTL200    ;PROCESS IF AVAILABLE
 1304   024C   CD   05   48         CALL ZGETKY    ;ANY KEYBOARD INPUT?
 1305   024F   CA   74   02         JZ   WTL200    ;YES - PROCESS IT
 1306   0252   .    .    .   ;
 1307   0252   .    .    .   ; IF KEYBOARD LOCKED, A = CHARACTER HIT, IF ANY
 1308   0252   .    .    .   ; UTHERWISE A = 377B
 1309   0252   .    .    .   ;
 1310   0252   FE   0D   .          CPI  CR        ;RETURN KEY HIT?
 1311   0254   C2   5F   02         JNZ  WTL020    ;NO - CHECK CTU & DISPATCHER
 1312   0257   3A   65   FF         LDA  IOFLGS    ;USER READ OR FILE READ
 1313   025A   E6   06   .          ANI  USREAD+FILRED  ;PENDING?
 1314   025C   C4   37   28         CNZ  RDABRT    ;YES - ABORT READ KEY
```

```
=========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                    PAGE  37
=========================================================================
 1316    025F   .    .    .   ;*****************************************
 1317    025F   .    .    .   ; CHECK CTU'S AND PENDING BLOCK TRANSFERS *
 1318    025F   .    .    .   ;*****************************************
 1319    025F   .    .    .   WTL020 EQU   $
 1320    025F   21   54   FF         LXI   H,SCNCNT  ;DECREMENT SCAN COUNT
 1321    0262   35   .    .          DCR   M         ;11 SCANS DONE?
 1322    0263   F2   36   02         JP    WTL010    ;NO - RESTART DO NOTHING LOO
 1323    0266   36   0A   .          MVI   M,10      ;YES - RESET SCAN COUNT
 1324    0268   CD   68   91         CALL  SCNVEC    ;DO OPTIONAL DISPLAY SCAN
 1325    026B   CD   86   15         CALL  IOCTMN    ;MONITOR TAPE DRIVES
 1326    026E   CD   29   04         CALL  DSPTCH    ;CHECK PENDING BLOCK XFRS
 1327    0271   C3   36   02         JMP   WTL010    ;RESTART DO NOTHING LOOP
```

```
=========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=========================================================================
 1329    0274   .    .    .   ;
 1330    0274   .    .    .   ;  KEY HIT - CHECK FOR FUNCTION KEY
 1331    0274   .    .    .   ;
 1332    0274   .    .    .   WTL200 EQU   $
 1333    0274   32   9C   FF         STA   CHARIN      ;SAVE KEYBOARD CHARACTER
 1334    0277   4F   .    .          MOV   C,A         ;SAVE THE BYTE IN C-REGISTER
 1335    0278   3E   FE   .          MVI   A,377Q-SDACOM
 1336    027A   CD   01   16         CALL  CLRDFL      ;CLEAR DATA COMM INPUT FLAG
 1337    027D   3A   F8   FF         LDA   CMFLGS      ;GET COMMON FLAGS
 1338    0280   2F   .    .          CMA               ;BOTH RECEIVE MODE FLAG SET
 1339    0281   E6   30   .          ANI   RCVMDE+REMSET  ;AND REMOTE ENABLED?
 1340    0283   C2   8C   02         JNZ   WTL205      ;NO - PROCESS KEYBOARD INPUT
 1341    0286   CD   14   48         CALL  ZBELL       ;YES - SOUND BELL AND
 1342    0289   C3   36   02         JMP   WTL010        ;IGNORE KEY
 1343    028C   .    .    .   ;
 1344    028C   .    .    .   WTL205 EQU   $
 1345    028C   AF   .    .          XRA   A           ;NO - PROCESS THE KEY
 1346    028D   57   .    .          MOV   D,A           ;(SET A,D = 0)
 1347    028E   B1   .    .          ORA   C           ;FUNCTION KEY?
 1348    028F   F2   F0   02         JP    WTL300      ;NO - PROCESS ASCII KEY
 1349    0292   FE   A1   .          CPI   FNCLIM      ;TABLE FUNCTION?
 1350    0294   F2   A6   02         JP    WTL210      ;NO - CHECK FOR F1-F8
 1351    0297   D6   98   .          SUI   FNCLWR      ;COMPUTE TABLE INDEX
 1352    0299   87   .    .          ADD   A
 1353    029A   5F   .    .          MOV   E,A         ;COMPUTE TABLE ADDRESS
 1354    029B   21   BC   14         LXI   H,FNCTAB      ;(D = 0)
 1355    029E   19   .    .          DAD   D
 1356    029F   CD   6D   19         CALL  CHAIN       ;GET THE FUNCTION ADDRESS
 1357    02A2   C7   .    .          RST   ;RSTJMP      GO PERFORM FUNCTION
 1358    02A3   C3   30   02         JMP   WTLOOP      ;RESTART WAIT LOOP
 1359    02A6   .    .    .   ;
 1360    02A6   .    .    .   ;  CHECK FOR F1-F8 KEY
 1361    02A6   .    .    .   ;
 1362    02A6   .    .    .   WTL210 EQU   $
 1363    02A6   FE   F0   .          CPI   F1CODE      ;IS THE KEY F1-F8?
 1364    02A8   DA   B9   02         JC    WTL250      ;NO - EXPAND ESCAPE SEQUENCE
 1365    02AB   FE   F8   .          CPI   F8CODE+1
 1366    02AD   D2   B9   02         JNC   WTL250      ;NO - EXPAND ESCAPE SEQUENCE
 1367    02B0   CD   8C   19         CALL  CHKSFK      ;SOFT KEY MODE?
 1368    02B3   CC   06   14         CZ    FCTKEY      ;NO - PROCESS FCT KEY
 1369    02B6   C3   30   02         JMP   WTLOOP      ;RESTART WAIT LOOP
```

```
====================================================================================
 1371   02B9   .   .   .   ;************************
 1372   02B9   .   .   .   ; PROCESS FUNCTION KEYS *
 1373   02B9   .   .   .   ;************************
 1374   02B9   .   .   .   WTL250 EQU  $
 1375   02B9   0E  1B  .          MVI  C,ESC      ;SET ESCAPE AS INPUT CHAR
 1376   02BB   .   .   .   WTL260 EQU  $
 1377   02BB   CD  E4  05         CALL LOCLIO     ;PROCESS KEYBOARD INPUT
 1378   02BE   21  9C  FF         LXI  H,CHARIN   ;RECALL KEYBOARD INPUT
 1379   02C1   7E  .   .          MOV  A,M
 1380   02C2   FE  FF  .          CPI  ENHNCF     ;DISPLAY ENHANCEMENT CODE?
 1381   02C4   CA  DC  02         JZ   WTL270     ;YES - EXPAND INTO AMPERSAND
 1382   02C7   FE  FE  .          CPI  STFOR1     ;ENTER FOREIGN MODE CONTROL?
 1383   02C9   CA  E3  02         JZ   WTL280     ;YES - CONTINUE SEQUENCE
 1384   02CC   FE  FD  .          CPI  STFOR2     ;COMPLETE FOREIGN MODE SET?
 1385   02CE   CA  E9  02         JZ   WTL290     ;YES - SET ENDING SEQUENCE
 1386   02D1   E6  7F  .          ANI  177Q       ;NO - MASK OUT UPPER BIT
 1387   02D3   BE  .   .          CMP  M          ;FUNCTION COMPLETED?
 1388   02D4   CA  30  02         JZ   WTLOOP     ;YES - RESTART WAIT LOOP
 1389   02D7   77  .   .          MOV  M,A        ;NO - SET NEW KEYBOARD CHAR
 1390   02D8   4F  .   .          MOV  C,A
 1391   02D9   C3  BB  02         JMP  WTL260     ;PERFORM THE DESIRED FUNCTON
 1392   02DC   .   .   .   ;
 1393   02DC   .   .   .   WTL270 EQU  $
 1394   02DC   36  E4  .          MVI  M,ESCLWD   ;SET <ESC>-<LOWER CASE D> AS
 1395   02DE   0E  26  .          MVI  C,AMPSND     ;CURRENT KEYBOARD CHARACTE
 1396   02E0   C3  BB  02         JMP  WTL260     ;PROCESS AMPERSAND
 1397   02E3   .   .   .   ;
 1398   02E3   .   .   .   WTL280 EQU  $
 1399   02E3   35  .   .          DCR  M          ;SET TO NEXT STEP CODE
 1400   02E4   0E  29  .          MVI  C,ARPARN   ;ENTER RIGHT PARENTHESIS
 1401   02E6   C3  BB  02         JMP  WTL260     ;PROCESS RIGHT PARENTHESIS
 1402   02E9   .   .   .   ;
 1403   02E9   .   .   .   WTL290 EQU  $
 1404   02E9   36  8E  .          MVI  M,ESCSO    ;SET <ESC>-<SO> AS CURRENT
 1405   02EB   0E  43  .          MVI  C,C          ;KEYBOARD CHARACTER
 1406   02ED   C3  BB  02         JMP  WTL260     ;PROCESS LETTER <C>
 1408   02F0    .    .    .   ;
 1409   02F0    .    .    .   ;  DISPLAYABLE CHARACTER - CHECK FOR APPROACHING
 1410   02F0    .    .    .   ;    END OF LINE WARNING
 1411   02F0    .    .    .   ;
 1412   02F0    .    .    .   WTL300 EQU  $
 1413   02F0   FE   20    .          CPI  CTLLIM      ;CONTROL CODE?
 1414   02F2   DA   13   03          JC   WTL310      ;YES - DON'T LOOK FOR BELL
 1415   02F5   3A   6E   FF          LDA  DFLGS       ;NO - GET DATA TRANSFER FLAG
 1416   02F8   E6   10    .          ANI  FCTK2D      ;PROCESSING FUNCTION KEY OR
 1417   02FA   CC   76   19          CZ   CHKFMS        ;FORMAT/SOFT KEY MODE?
 1418   02FD   C2   13   03          JNZ  WTL310      ;YES - SKIP BELL COLUMN CHEC
 1419   0300   3A   D1   FF          LDA  ESCFLG      ;NO - GET ESCAPE SEQ FLAG
 1420   0303   B7    .    .          ORA  A           ;CURRENTLY IN ESCAPE SEQ?
 1421   0304   C2   13   03          JNZ  WTL310      ;YES - DON'T LOOK FOR BELL
 1422   0307   3A   C1   FF          LDA  CURCOL      ;NO - GET CURRENT COLUMN
 1423   030A   21   BE   FF          LXI  H,RHTMGN    ;COMPARE TO RIGHT MARGIN
 1424   030D   96    .    .          SUB  M           ;CLOSE ENOUGH TO RIGHT MARGI
 1425   030E   C6   08    .          ADI  BELLIM        ;TO SOUND BELL?
 1426   0310   CC   14   48          CZ   ZBELL       ;YES - SOUND BELL
 1427   0313    .    .    .   ;***************************
 1428   0313    .    .    .   ; PROCESS THE KEY FUNCTION *
 1429   0313    .    .    .   ;***************************
 1430   0313    .    .    .   WTL310 EQU  $
 1431   0313   CD   EF   05          CALL LOCLIN      ;PERFORM LOCAL INPUT ROUTINE
 1432   0316   3A   9C   FF          LDA  CHARIN      ;RECALL KEYBOARD INPUT CHAR
 1433   0319   FE   0D    .          CPI  CR          ;WAS IT A RETURN?
 1434   031B   C2   30   02          JNZ  WTLOOP      ;NO - RESTART WAIT LOOP
 1435   031E   3A   F3   FF          LDA  MDFLG2      ;YES - GET MODE FLAGS
 1436   0321   E6   04    .          ANI  AUTOLF      ;AUTO LINE FEED ENABLED?
 1437   0323   CA   30   02          JZ   WTLOOP      ;NO - RESTART WAIT LOOP
 1438   0326   2E   01    .          MVI  L,1         ;YES - DELAY 10 MILLISECONDS
 1439   0328   CD   B3   12          CALL DELAY         ;THEN SEND LINE FEED
 1440   032B   3E   0A    .          MVI  A,LF
 1441   032D   C3   74   02          JMP  WTL200      ;FAKE LINE FEED FROM KEYBOAR
```

```
=========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                    PAGE  41
=========================================================================
 1443   0330   .    .    .   ;************************
 1444   0330   .    .    .   ; S U B R O U T I N E S *
 1445   0330   .    .    .   ;************************
 1446   0330   .    .    .   ;
 1447   0330   .    .    .   ; * * * * * * * * * * * * * * * * * * * * * * * *
 1448   0330   .    .    .   ;
 1449   0330   .    .    .   ;  CHINT - INTERPRET INPUT CHARACTER
 1450   0330   .    .    .   ;
 1451   0330   .    .    .   ;      ENTRY:  C = INPUT CHARACTER
 1452   0330   .    .    .   ;
 1453   0330   .    .    .   ;      EXIT :  Z - FAST STORE USED
 1454   0330   .    .    .   ;             NZ - FULL PROCESSING USED
 1455   0330   .    .    .   ;             A-E,L DESTROYED
 1456   0330   .    .    .   ;
 1457   0330   .    .    .   ;  TRY FAST STORE FIRST
 1458   0330   .    .    .   ;
 1459   0330   .    .    .   CHINT0 EQU  $          ;ENTRY FOR I/O INPUT
 1460   0330   21   6F   FF         LXI  H,MFLGS2   ;SET H,L TO MODE FLAGS 2
 1461   0333   79   .    .          MOV  A,C        ;PUT INPUT CHAR IN A-REG
 1462   0334   FE   0A   .          CPI  LF         ;CHARACTER = LINE FEED?
 1463   0336   C2   42   03         JNZ  CHI000     ;NO - CHECK FOR CR/DC3
 1464   0339   7E   .    .          MOV  A,M        ;YES - GET MODE FLAGS 2
 1465   033A   F6   40   .          ORI  WRPFLG     ;TURN ON WRAP FLAG
 1466   033C   BE   .    .          CMP  M          ;WRAP FLAG ALREADY ON?
 1467   033D   CA   50   03         JZ   CHINT      ;YES - EXECUTE LINE FEED
 1468   0340   77   .    .          MOV  M,A        ;NO - SET WRAP FLAG
 1469   0341   C9   .    .          RET               ;AND IGNORE LINE FEED
 1470   0342   .    .    .   ;
 1471   0342   .    .    .   CHI000 EQU  $
 1472   0342   FE   0D   .          CPI  CR         ;CHARACTER = RETURN?
 1473   0344   CA   B9   03         JZ   CHINT1     ;YES - DON'T SET WRAP FLAG
 1474   0347   FE   13   .          CPI  DC3        ;CHARACTER = DC3?
 1475   0349   CA   50   03         JZ   CHINT      ;YES - DON'T SET WRAP FLAG
 1476   034C   7E   .    .          MOV  A,M        ;NO - SET WRAP FLAG
 1477   034D   F6   40   .          ORI  WRPFLG
 1478   034F   77   .    .          MOV  M,A        ;UPDATE MODE FLAGS 2
 1479   0350   .    .    .   CHINT  EQU  $
 1480   0350   21   67   FF         LXI  H,CRAFLG
 1481   0353   46   .    .          MOV  B,M        ;WAS LAST CHARACTER FUNCTION
 1482   0354   05   .    .          DCR  B            ;A CURSOR ADVANCE?
 1483   0355   FA   B9   03         JM   CHI100     ;NO - DO FULL PROCESSING
 1484   0358   70   .    .          MOV  M,B        ;YES - CLEAR FLAG
 1485   0359   79   .    .          MOV  A,C        ;PUT INPUT CHARACTER IN A-RE
 1486   035A   FE   20   .          CPI  CTLLIM     ;IS CHARACTER A CONTROL CODE
 1487   035C   FA   A8   03         JM   CHI050     ;YES - CHECK FOR DISPLAY FCT
 1488   035F   .    .    .   ;                      NO - DO FAST STORE
 1490   035F    .   .   .   ;
 1491   035F    .   .   .   ;  FAST STORE PROCESSING
 1492   035F    .   .   .   ;
 1493   035F    .   .   .   CHI010 EQU  $
 1494   035F    2E  6C  .          MVI  L,SPOWL-BASE
 1495   0361    46  .   .          MOV  B,M          ;GET THE SPOW LATCH IN B-REG
 1496   0362    2A  C3  FF         LHLD CURADR       ;GET LAST CHAR DONE ADDRESS
 1497   0365    7E  .   .          MOV  A,M          ;GET LAST CHARACTER DONE
 1498   0366    B7  .   .          ORA  A            ;IS IT ASCII?
 1499   0367    FA  B9  03         JM   CHI100       ;NO - DO FULL PROCESSING
 1500   036A    2B  .   .          DCX  H            ;YES - GET NEXT CHARACTER
 1501   036B    7E  .   .          MOV  A,M
 1502   036C    B7  .   .          ORA  A            ;IS IT ASCII?
 1503   036D    F2  8D  03         JP   CHI020       ;YES - OVERLAY EXISTING CHAR
 1504   0370    FE  CC  .          CPI  EOL          ;IS IT EOL?
 1505   0372    C2  B9  03         JNZ  CHI100       ;NO - DO FULL PROCESSING
 1506   0375    47  .   .          MOV  B,A          ;YES - SAVE EOL AND CLEAR
 1507   0376    2B  .   .          DCX  H              ;SPOW LATCH COMPARE
 1508   0377    7E  .   .          MOV  A,M          ;GET NEXT CHARACTER
 1509   0378    FE  C3  .          CPI  FILL         ;IS IT AN END OF LINE FILL?
 1510   037A    C2  B9  03         JNZ  CHI100       ;NO - DO FULL PROCESSING
 1511   037D    3A  C0  FF         LDA  CURROW       ;YES - ADD CHAR TO DISPLAY
 1512   0380    F6  40  .          ORI  MAYEOL       ;SET DMA OFF WITH EOL SKIP
 1513   0382    F3  .   .          DI                ;DISABLE INTERRUPTS
 1514   0383    32  20  87         STA  IOCRRW       ;TURN OFF DMA
 1515   0386    3E  04  .          MVI  A,RSTOFF     ;DISABLE RESET KEY
 1516   0388    32  80  83         STA  IOKBCO
 1517   038B    70  .   .          MOV  M,B          ;STORE NEW EOL
 1518   038C    23  .   .          INX  H            ;SET TO OLD EOL ADDRESS
 1520   038D    .    .    .   ;
 1521   038D    .    .    .   ;  ADD CHARACTER TO DISPLAY
 1522   038D    .    .    .   ;
 1523   038D    .    .    .   CHI020 EQU  $
 1524   038D    79   .    .          MOV  A,C         ;RECALL THE INPUT CHARACTER
 1525   038E    B8   .    .          CMP  B           ;STORE INHIBITED BY SPOW?
 1526   038F    CA   93   03         JZ   CHI030      ;YES - BYPASS STORE
 1527   0392    71   .    .          MOV  M,C         ;NO - STORE THE BYTE
 1528   0393    .    .    .   CHI030 EQU  $
 1529   0393    CD   9E   0F         CALL DISLN1      ;TURN DISPLAY BACK ON
 1530   0396    22   C3   FF         SHLD CURADR      ;STORE NEW CURRENT ADDRESS
 1531   0399    21   C8   FF         LXI  H,LSTCOL    ;INCREMENT LSTCOL
 1532   039C    34   .    .          INR  M
 1533   039D    CD   05   20         CALL CURADV      ;ADVANCE CURSOR
 1534   03A0    .    .    .   ;****************************************
 1535   03A0    .    .    .   ; CHINT2 - SET CURSOR COLUMN ON DISPLAY *
 1536   03A0    .    .    .   ;****************************************
 1537   03A0    .    .    .   ;
 1538   03A0    .    .    .   ;  EXIT :  Z TRUE
 1539   03A0    .    .    .   ;          A DESTROYED
 1540   03A0    .    .    .   ;
 1541   03A0    .    .    .   CHINT2 EQU  $
 1542   03A0    BF   .    .          CMP  A
 1543   03A1    3A   C1   FF         LDA  CURCOL      ;GET CURRENT COLUMN NUMBER
 1544   03A4    32   00   87         STA  IOCRCL      ;SET DISPLAY CURSOR COLUMN
 1545   03A7    C9   .    .          RET              ;RETURN
 1546   03A8    .    .    .   ;
 1547   03A8    .    .    .   ;  CONTROL CODE - CHECK FOR DISPLAY FUNCTIONS
 1548   03A8    .    .    .   ;
 1549   03A8    .    .    .   CHI050 EQU  $
 1550   03A8    CD   47   10         CALL CKDSPF      ;DISPLAY FUNCTIONS ENABLED?
 1551   03AB    CA   B9   03         JZ   CHI100      ;NO - DO FULL PROCESSING
 1552   03AE    79   .    .          MOV  A,C         ;YES - RECALL INPUT CHARACTE
 1553   03AF    FE   0D   .          CPI  CR          ;IS IT RETURN CHARACTER?
 1554   03B1    CA   B9   03         JZ   CHI100      ;YES - DO FULL PROCESSING
 1555   03B4    FE   1B   .          CPI  ESC         ;IT IT AN ESCAPE?
 1556   03B6    C2   5F   03         JNZ  CHI010      ;NO - DO FAST PROCESSING
 1557   03B9    .    .    .   ;                        YES - DO FULL PROCESSING
 1559    03B9    .    .    .    ;
 1560    03B9    .    .    .    ;  FULL PROCESSING
 1561    03B9    .    .    .    ;
 1562    03B9    .    .    .    CHINT1 EQU   $
 1563    03B9    .    .    .    CHI100 EQU   $
 1564    03B9    61   .    .           MOV   H,C
 1565    03BA    69   .    .           MOV   L,C         ;SET "CHAR" AND "DCHAR" TO
 1566    03BB    22   88   FF          SHLD  CHAR          ;CURRENT CHARACTER
 1567    03BE    CD   23   20          CALL  CRADV1      ;CLEAR CURADV FLAG
 1568    03C1    .    .    .    ;********************************
 1569    03C1    .    .    .    ; DETERMINE CHARACTER FUNCTION *
 1570    03C1    .    .    .    ;********************************
 1571    03C1    2A   D2   FF          LHLD  RNGTA       ;GET CURRENT RANGE TABLE ADD
 1572    03C4    .    .    .    ;******************************
 1573    03C4    .    .    .    ; ADVANCE TO NEXT TABLE ENTRY *
 1574    03C4    .    .    .    ;******************************
 1575    03C4    .    .    .    CHI110 EQU   $
 1576    03C4    23   .    .           INX   H
 1577    03C5    23   .    .           INX   H
 1578    03C6    23   .    .           INX   H
 1579    03C7    .    .    .    ;*************************************
 1580    03C7    .    .    .    ; COMPARE CHARACTER TO CURRENT RANGE *
 1581    03C7    .    .    .    ;*************************************
 1582    03C7    79   .    .           MOV   A,C         ;PUT CHARACTER IN A-REGISTER
 1583    03C8    96   .    .           SUB   M           ;CHARACTER >= LOWER BOUND?
 1584    03C9    23   .    .           INX   H             ;(SET H,L TO UPPER BOUND)
 1585    03CA    DA   C4   03          JC    CHI110      ;NO - ADVANCE TO NEXT ENTRY
 1586    03CD    07   .    .           RLC               ;YES - DOUBLE DIFFERENCE AND
 1587    03CE    47   .    .           MOV   B,A           ;SAVE VALUE IN B-REGISTER
 1588    03CF    7E   .    .           MOV   A,M         ;GET UPPER BOUND
 1589    03D0    B9   .    .           CMP   C           ;CHARACTER <= UPPER BOUND?
 1590    03D1    DA   C4   03          JC    CHI110      ;NO - ADVANCE TO NEXT ENTRY
 1591    03D4    .    .    .    ;***********************************************
 1592    03D4    .    .    .    ; CHARACTER FUNCTION FOUND - GET FUNCTION ADDR *
 1593    03D4    .    .    .    ;***********************************************
 1594    03D4    23   .    .           INX   H
 1595    03D5    5E   .    .           MOV   E,M         ;PUT ADDRESS ENTRY IN
 1596    03D6    23   .    .           INX   H             ;A (= MSB), E (= LSB)
 1597    03D7    7E   .    .           MOV   A,M
 1598    03D8    E6   7F   .           ANI   177Q        ;MASK OUT HIGH ORDER BIT
 1599    03DA    57   .    .           MOV   D,A           ;(PUT NEw MSB INTO D-REG)
 1600    03DB    96   .    .           SUB   M           ;USE INDEX TABLE?
 1601    03DC    C2   E5   03          JNZ   CHI200      ;NO - USE AS FUNCTON ADDRESS
 1602    03DF    68   .    .           MOV   L,B         ;YES - PUT DIFFERENCE IN H,L
 1603    03E0    67   .    .           MOV   H,A           ;(A = 0)
 1604    03E1    19   .    .           DAD   D           ;COMPUTE TABLE ADDRESS
 1605    03E2    5E   .    .           MOV   E,M         ;GET INDEX TABLE VALUE
 1606    03E3    23   .    .           INX   H
 1607    03E4    56   .    .           MOV   D,M
```

```
1609   03E5   .   .   .    ;*****************************
1610   03E5   .   .   .    ; PERFORM CHARACTER FUNCTION *
1611   03E5   .   .   .    ;*****************************
1612   03E5   .   .   .    CHI200 EQU  $
1613   03E5   EB  .   .           XCHG
1614   03E6   22  CE  FF          SHLD CNTFAD    ;SET FUNCTION ADDRESS
1615   03E9   06  01  .           MVI  B,1       ;SET INITIAL FUNCTION INDEX
1616   03EB   26  FF  .           MVI  H,BASEH   ;SET H TO DATA PAGE
1617   03ED   3E  04  .           MVI  A,RSTOFF  ;DISABLE RESET KEY
1618   03EF   32  80  83          STA  IOKBCO
1619   03F2   .   .   .    ;**********************************************
1620   03F2   CD  CD  FF          CALL ECONTF    ;EXECUTE CHARACTER FUNCTION
1621   03F5   .   .   .    ;**********************************************
1622   03F5   CD  A4  0F          CALL DISLN3    ;RE-ENABLE RESET KEY
1623   03F8   CD  47  10          CALL CKDSPF    ;DISPLAY FUNCTIONS ENABLED?
1624   03FB   C2  0A  04          JNZ  CHI270    ;YES - DON'T END ESCAPE SEQ'
1625   03FE   21  D1  FF          LXI  H,ESCFLG  ;NO - CHECK ESCAPE FLAG
1626   0401   46  .   .           MOV  B,M
1627   0402   05  .   .           DCR  B         ;ESCAPE SEQUENCE IN PROGRESS
1628   0403   FA  0A  04          JM   CHI270    ;NO - DON'T CHANGE ESC FLAG
1629   0406   70  .   .           MOV  M,B       ;YES - UPDATE ESCAPE COUNTER
1630   0407   CC  95  04          CZ   ESCEND    ;RESET RANGE TABLE POINTER
1631   040A   .   .   .    ;                       COUNTER BECAME ZERO
1632   040A   .   .   .    CHI270 EQU  $
1633   040A   3A  88  FF          LDA  CHAR      ;SAVE THE LAST CHARACTER
1634   040D   32  69  FF          STA  LCHAR       ;PROCESSED
1635   0410   BC  .   .           CMP  H         ;SET Z FALSE
1636   0411   C9  .   .           RET            ;RETURN
```

```
 1638   0412    .    .    .  ;*********************************************
 1639   0412    .    .    .  ; CHECK CONTROL CODES FOR BLOCK TERMINATOR OR *
 1640   0412    .    .    .  ;   BLOCK TRANSFER TRIGGER                    *
 1641   0412    .    .    .  ;*********************************************
 1642   0412    .    .    .  ;
 1643   0412    .    .    .  ;  ENTRY:  C = INPUT CHARACTER
 1644   0412    .    .    .  ;
 1645   0412    .    .    .  CHKCTL EQU  $
 1646   0412    3A   04   50        LDA  BLKTRM     ;GET BLOCK TERMINATOR CHAR
 1647   0415    B9   .    .         CMP  C          ;INPUT = BLOCK TERMINATOR?
 1648   0416    CA   BA   0C        JZ   SFKYDS     ;YES - DISPLAY INPUT
 1649   0419    3A   6E   FF        LDA  DFLGS      ;GET TRANSFER FLAGS
 1650   041C    E6   01   .         ANI  SDACOM     ;INPUT FROM DATA COMM?
 1651   041E    C8   .    .         RZ              ;NO - DO NOTHING
 1652   041F    3A   02   50        LDA  TRIGGR     ;IS INPUT CHARACTER THE
 1653   0422    B9   .    .         CMP  C            ;BLOCK TRANSFER TRIGGER?
 1654   0423    C0   .    .         RNZ             ;NO - DO NOTHING
 1655   0424    .    .    .  ;
 1656   0424    .    .    .  CHKCT1 EQU  $          ;SET BLOCK TRANSFER TRIGGER
 1657   0424    3E   01   .         MVI  A,SETTRG   ;GO TO DATA COMM ROUTINE TO
 1658   0426    C3   42   12        JMP  DCMCTL       ;SET BLOCK TRANSFER TRIGGE
 1660   0429   .   .   .    ;*******************************************
 1661   0429   .   .   .    ; DSPTCH - DISPATCH PENDING BLOCK TRANSFERS *
 1662   0429   .   .   .    ;*******************************************
 1663   0429   .   .   .    DSPTCH EQU  $
 1664   0429   3A  F8  FF          LDA  CMFLGS     ;GET COMMON FLAGS
 1665   042C   E6  01  .           ANI  BLKTRG     ;BLOCK TRANSFER TRIGGER SET?
 1666   042E   C8  .   .           RZ              ;NO - RETURN
 1667   042F   3A  70  FF          LDA  MFLGS      ;YES - RELEASE ANY PENDING
 1668   0432   21  51  04          LXI  H,DSPTAB     ;BLOCK TRANSFERS
 1669   0435   0E  08  .           MVI  C,NMPNDG
 1670   0437   .   .   .    ;
 1671   0437   .   .   .    DSP010 EQU  $
 1672   0437   0F  .   .           RRC             ;TRANSFER PENDING BIT SET?
 1673   0438   DA  4D  04          JC   DSP020     ;YES - GO DO TRANSFER
 1674   043B   23  .   .           INX  H          ;NO - CHECK NEXT BIT
 1675   043C   23  .   .           INX  H          ;INCREMENT FUNCTION TABLE AD
 1676   043D   0D  .   .           DCR  C          ;ALL BITS CHECKED?
 1677   043E   C2  37  04          JNZ  DSP010     ;NO - CONTINUE CHECKING
 1678   0441   .   .   .    ;
 1679   0441   3A  6F  FF          LDA  MFLGS2     ;YES - CHECK 2ND SET OF FLAG
 1680   0444   0F  .   .           RRC             ;DEVICE RECORD PENDING?
 1681   0445   DA  23  28          JC   IORDGO     ;YES - SEND I/O RECORD
 1682   0448   0F  .   .           RRC             ;BINARY DATA PENDING?
 1683   0449   DA  29  28          JC   BNRYGO     ;YES - TRANSMIT THE DATA
 1684   044C   C9  .   .           RET             ;NO - RETURN
 1685   044D   .   .   .    ;********************************************
 1686   044D   .   .   .    ; PENDING BIT FOUND - GO TO TRANSMIT FUNCTION *
 1687   044D   .   .   .    ;********************************************
 1688   044D   .   .   .    DSP020 EQU  $
 1689   044D   CD  6D  19          CALL CHAIN      ;GET TRANSMIT FUNCITON ADDR
 1690   0450   E9  .   .           PCHL            ;GO TO THE FUNCTION
 1691   0451   .   .   .    ;
 1692   0451   .   .   .    DSPTAB EQU  $
 1693   0451   28  12  .           DW   DC2GO      ;SEND DC2
 1694   0453   F9  0C  .           DW   STATGO     ;SEND TERMINAL STATUS
 1695   0455   12  26  .           DW   STA2GO     ;SEND TERMINAL STATUS 2
 1696   0457   1D  28  .           DW   IOSTGO     ;SEND I/O STATUS
 1697   0459   E4  11  .           DW   CPSNGO     ;SEND CURSOR ADDRESS
 1698   045B   51  14  .           DW   FKEYGO     ;SEND FUNCTION KEY DATA
 1699   045D   3B  13  .           DW   DPSGO      ;SEND DISPLAY DATA
 1700   045F   20  28  .           DW   IODNGO     ;SEND I/O TERMINATION CODE
 1701   0461   .   .   .    ;
 1702   0008   .   .   .    NMPNDG EQU  ($-DSPTAB)/2
```

```
0461  .  .  .   ;******************************
0461  .  .  .   ; ESCAPE CHARACTER PROCESSING *
0461  .  .  .   ;******************************
0461  .  .  .   ESCAPE EQU $
0461  3A 6E FF         LDA  DFLGS
0464  E6 01 .          ANI  SDACOM    ;DATA FROM DATACOM?
0466  CA 73 04         JZ   ESC010    ;NO - DON'T LOCK KEYBOARD
0469  3A F3 FF         LDA  MDFLG2    ;YES - GET MODE FLAGS
046C  E6 02 .          ANI  BLKMDE    ;BLOCK MODE?
046E  3E 08 .          MVI  A,ESCINP    ;(PUT IGNORE FLAG IN A-REG
0470  C4 39 17         CNZ  SETMF2    ;YES - SET IGNORE KEYBD FLAG
0473  .  .  .   ESC010 EQU  $
0473  CD 8C 19         CALL CHKSFK    ;SOFT KEY MODE?
0476  21 9A 26         LXI  H,ESCTAB    ;(SET FOR NORMAL ESC TABLE
0479  CA 81 04         JZ   ESCAPO    ;NO - SET RANGE TABLE
047C  21 86 26         LXI  H,SESCTB  ;YES - USE SOFT KEY TABLE
047F  .  .  .   ;**********************************************
047F  .  .  .   ; ESCAPO - SET RANGE TABLE FOR ESCAPE SEQUENCE *
047F  .  .  .   ;**********************************************
047F  .  .  .   ;
047F  .  .  .   ;  ENTRY:  A = RADIX (BASE) FOR DIGIT PARAMETERS
047F  .  .  .   ;          H,L = ADDRESS OF NEW RANGE TABLE
047F  .  .  .   ;
047F  .  .  .   ;  EXIT :  H,L = ESCFLG
047F  .  .  .   ;
047F  .  .  .   ;  ESCAPA - USE DECIMAL RADIX
047F  .  .  .   ;
047F  .  .  .   ESCAPA EQU  $
047F  3E 0A .          MVI  A,DECRDX  ;SET RADIX FOR BASE 10 DIGIT
0481  .  .  .   ESCAPO EQU  $
0481  32 D4 FF         STA  RADIX     ;SET PARAMETER RADIX
0484  22 D2 FF         SHLD RNGTA     ;SET NEW RANGE TABLE
0487  .  .  .   ESCAPB EQU  $         ;ENTRY TO CLEAR ACCUMULATOR
0487  21 DD FF         LXI  H,IOCSGN  ;CLEAR OUT THE PARAMETER
048A  1E 03 .          MVI  E,3         ;ACCUMULATOR AREA
048C  CD FF 10         CALL CLRAL1
048F  .  .  .   ESCAP1 EQU  $
048F  21 D1 FF         LXI  H,ESCFLG  ;SET FLAG TO RESET AFTER
0492  36 02 .          MVI  M,2         ;FOLLOWING CHARACTER
0494  C9 .  .          RET            ;RETURN
```

```
1745   0495   .    .    .    ;*********************************************
1746   0495   .    .    .    ; ESCEND - END OF ESCAPE SEQUENCE PROCESSING *
1747   0495   .    .    .    ;*********************************************
1748   0495   .    .    .    ESCEND EQU  $
1749   0495   21   64   26         LXI  H,RTABLE  ;SET FOR NORMAL RANGE TABLE
1750   0498   CD   8C   19         CALL CHKSFK    ;SOFT KEY MODE?
1751   049B   CA   A1   04         JZ   ESCEN1    ;NO - USE NORMAL TABLE
1752   049E   21   60   26         LXI  H,DFST80  ;YES - USE SOFT KEY TABLE
1753   04A1   .    .    .    ESCEN1 EQU  $
1754   04A1   22   D2   FF         SHLD RNGTA        ;RESET RANGE TABLE POINTER
1755   04A4   AF   .    .          XRA  A            ;CLEAR ESCAPE FLAG AND
1756   04A5   32   D1   FF         STA  ESCFLG        ;ESCAPE KEYBOARD LOCKOUT
1757   04A8   3E   F7   .          MVI  A,377Q-ESCINP  ;FLAG
1758   04AA   .    .    .    ;*************************************
1759   04AA   .    .    .    ; CLRMF2 - CLEAR FLAG BIT IN MFLGS2 *
1760   04AA   .    .    .    ;*************************************
1761   04AA   .    .    .    ;
1762   04AA   .    .    .    ;  ENTRY:  A = 377B - FLAG BIT TO BE CLEARED
1763   04AA   .    .    .    ;
1764   04AA   .    .    .    ;  EXIT :  A = UPDATED MFLGS2 VALUE
1765   04AA   .    .    .    ;          H,L = MFLGS2
1766   04AA   .    .    .    ;
1767   04AA   .    .    .    CLRMF2 EQU  $
1768   04AA   21   6F   FF         LXI  H,MFLGS2
1769   04AD   A6   .    .          ANA  M            ;CLEAR THE FLAG BIT
1770   04AE   77   .    .          MOV  M,A          ;STORE NEW SETTINGS
1771   04AF   C9   .    .          RET               ;RETURN
```

```
04B0     .    .    .      ;
04B0     .    .    .      ; * * * * * * * * * * * * * * * * * * * * * * * *
04B0     .    .    .      ;
04B0     .    .    .      ;  FNDRAM - LOCATE END OF RAM SPACE
04B0     .    .    .      ;
04B0     .    .    .      ;      ENTRY:  B = MSB OF RAM SPACE LOWER LIMIT
04B0     .    .    .      ;              H,L = ADDR OF UPPER BOUNDARY
04B0     .    .    .      ;
04B0     .    .    .      ;      EXIT :  H,L = ADDRESS OF LOWER BOUNDARY
04B0     .    .    .      ;              A DESTROYED
04B0     .    .    .      ;
04B0     .    .    .      FNDRAM EQU  $
04B0     AF   .    .             XRA  A
04B1     6F   .    .             MOV  L,A          ;SET ADDRESS'S LSB TO ZERO
04B2     77   .    .             MOV  M,A          ;SET RAM LOCATION TO ZERO
04B3     BE   .    .             CMP  M            ;ALL ZEROES STORED?
04B4     C2   C4   04            JNZ  FRM010       ;NO - RAM LIMIT FOUND
04B7     35   .    .             DCR  M            ;YES - TRY TO SET TO ALL ONE
04B8     34   .    .             INR  M            ;ALL ONES STORED?
04B9     C2   C4   04            JNZ  FRM010       ;NO - RAM LIMIT FOUND
04BC     7C   .    .             MOV  A,H          ;YES - MOVE TO NEXT 1K
04BD     D6   04   .             SUI  4
04BF     67   .    .             MOV  H,A
04C0     B8   .    .             CMP  B            ;RAM LIMIT REACHED?
04C1     F2   B0   04            JP   FNDRAM       ;NO - TRY NEXT 1K
04C4     .    .    .      ;
04C4     .    .    .      ;  RAM LIMIT FOUND - RETURN LOW BOUNDARY
04C4     .    .    .      ;
04C4     .    .    .      FRM010 EQU $
04C4     7C   .    .             MOV  A,H          ;ADJUST H,L TO TRUE LOWER
04C5     C6   04   .             ADI  4              ;BOUNDARY
04C7     E6   FC   .             ANI  374Q         ;MASK FOR 1K START ADDRESS
04C9     67   .    .             MOV  H,A
04CA     C9   .    .             RET               ;RETURN
```

```
1808   04CB    .    .    .   ;
1809   04CB    .    .    .   ; * * * * * * * * * * * * * * * * * * * * * * * * *
1810   04CB    .    .    .   ;
1811   04CB    .    .    .   ;  GETBUF - GET BUFFER SPACE
1812   04CB    .    .    .   ;
1813   04CB    .    .    .   ;        ENTRY:  B,C = LENGTH OF BUFFER REQUIRED
1814   04CB    .    .    .   ;
1815   04CB    .    .    .   ;        EXIT :  A,H,L DESTROYED
1816   04CB    .    .    .   ;                P - BUFFER SPACE ALLOCATAED
1817   04CB    .    .    .   ;                  D,E = BUFFER START ADDRESS
1818   04CB    .    .    .   ;                M - BUFFER SPACE NOT ALLOCATED
1819   04CB    .    .    .   ;                  D,E DESTROYED
1820   04CB    .    .    .   ;
1821   04CB    .    .    .   ;  THIS ROUTINE ALLOCATES A CONTIGUOUS AREA OF
1822   04CB    .    .    .   ;    RAM.  THE BUFFER SPACE MAY NOT START ON A
1823   04CB    .    .    .   ;    256 BYTE PAGE BOUNDARY.
1824   04CB    .    .    .   ;
1825   04CB    .    .    .   GETBUF EQU $
1826   04CB    2A   8B   FF         LHLD BUFEND     ;GET CURRENT BUFFER END AND
1827   04CE    11   8E   FF         LXI  D,BUFBGN+1  ;ADDRESS OF BEGIN PTR'S MS
1828   04D1    CD   F2   04         CALL GTB100     ;ENOUGH SPACE?
1829   04D4    FA   DD   04         JM   GTB010     ;NO - TRY DISPLAY AREA
1830   04D7    22   8B   FF         SHLD BUFEND     ;YES - STORE NEW BUFFER END
1831   04DA    .    .    .   GTB005 EQU  $
1832   04DA    EB   .    .          XCHG            ;SET D,E TO LOW ADDRESS
1833   04DB    13   .    .          INX  D            ;OF BUFFER
1834   04DC    C9   .    .          RET             ;RETURN
1835   04DD    .    .    .   ;
1836   04DD    .    .    .   ;  NOT ENOUGH NON-DISPLAY RAM - TRY DISPLAY AREA
1837   04DD    .    .    .   ;
1838   04DD    .    .    .   GTB010 EQU  $
1839   04DD    2A   A8   FF         LHLD DSPEND     ;GET CURRENT DISPLAY END AND
1840   04E0    11   AB   FF         LXI  D,DSPBGN+1  ;ADDR OF BEGIN PTR'S MSB
1841   04E3    CD   F2   04         CALL GTB100     ;ENOUGH SPACE?
1842   04E6    22   A8   FF         SHLD DSPEND       ;(STORE NEW DISPLAY END)
1843   04E9    F2   DA   04         JP   GTB005     ;YES - RETURN BUFFER ADDRESS
1844   04EC    21   27   0F         LXI  H,BUFMSG   ;NO - REPORT ERROR
1845   04EF    C3   54   12         JMP  HANG00
1846   04F2    .    .    .   ;
1847   04F2    .    .    .   ;  GTB100 - CHECK FOR AVAILABLE SPACE
1848   04F2    .    .    .   ;
1849   04F2    .    .    .   GTB100 EQU  $
1850   04F2    7D   .    .          MOV  A,L        ;SUBTRACT DESIRED SPACE
1851   04F3    91   .    .          SUB  C            ;FROM END OF REGION
1852   04F4    6F   .    .          MOV  L,A
1853   04F5    7C   .    .          MOV  A,H
1854   04F6    98   .    .          SBB  B
1855   04F7    67   .    .          MOV  H,A
1856   04F8    EB   .    .          XCHG            ;COMPARE NEW MSB OF END TO
1857   04F9    BE   .    .          CMP  M            ;MSB OF BEGINNING
1858   04FA    EB   .    .          XCHG            ;PUT NEW END ADDRESS IN H,L
1859   04FB    C9   .    .          RET             ;RETURN (P = ENOUGH)
```

```
1861    04FC   .   .   .   ;*****************************************
1862    04FC   .   .   .   ; GETDCM - PROCESS DATA COMM INPUT IF ANY *
1863    04FC   .   .   .   ;*****************************************
1864    04FC   .   .   .   ;
1865    04FC   .   .   .   ;   ENTRY:  DON'T CARE
1866    04FC   .   .   .   ;
1867    04FC   .   .   .   ;   EXIT :  NC
1868    04FC   .   .   .   ;           NZ - DATA COMM INPUT BUFFER EMPTY
1869    04FC   .   .   .   ;           Z - EXIT ON FULL INPUT PROCESSING
1870    04FC   .   .   .   ;           ALL REGISTERS DESTROYED
1871    04FC   .   .   .   ;
1872    04FC   .   .   .   GETDCM EQU  $
1873    04FC   3A  F3  FF         LDA  MDFLG2      ;GET HARD MODE FLAGS
1874    04FF   E6  08  .          ANI  REMOTE      ;REMOTE MODE ENABLED?
1875    0501   3A  F8  FF         LDA  CMFLGS        ;(GET COMMON FLAGS)
1876    0504   CA  44  05         JZ   GDC100      ;NO - IGNORE DATA COMM
1877    0507   F6  10  .          ANI  REMSET      ;WAS REMOTE ON BEFORE?
1878    0509   CC  E2  13         CZ   ENTREM      ;NO - SET REMOTE MODE
1879    050C   .   .   .   ;**********************
1880    050C   .   .   .   ; GET DATA COMM INPUT *
1881    050C   .   .   .   ;**********************
1882    050C   .   .   .   GDC010 EQU  $
1883    050C   CD  17  50         CALL ZGETDC      ;ANY DATA COMM INPUT?
1884    050F   DA  36  05         JC   GDC050        ;(PROCESS ERROR IF ANY)
1885    0512   C0  .   .          RNZ              ;NO - RETURN
1886    0513   4F  .   .          MOV  C,A         ;YES - SAVE INPUT INTO C-REG
1887    0514   .   .   .   ;***************************
1888    0514   .   .   .   ; PROCESS DATA COMM INPUT *
1889    0514   .   .   .   ;***************************
1890    0514   .   .   .   GDC020 EQU  $
1891    0514   CD  0F  17         CALL SETDFO      ;SET DATA COMM INPUT FLAG
1892    0517   3A  F4  FF         LDA  MDFLG1      ;GET SOFT MODE FLAGS
1893    051A   E6  40  .          ANI  RECORD      ;RECORD MODE ENABLED?
1894    051C   CA  2A  05         JZ   GDC030      ;NO - PROCESS THE INPUT
1895    051F   79  .   .          MOV  A,C         ;YES - LOOK FOR RECORD TRIGG
1896    0520   FE  0D  .          CPI  CR          ;INPUT = RETURN?
1897    0522   CA  2A  05         JZ   GDC030      ;YES - PROCESS THE CHARACTER
1898    0525   FE  0A  .          CPI  LF          ;IS IT LINE FEED?
1899    0527   C2  26  28         JNZ  RCRDGO      ;NO - EXECUTE RECORD FUNCTIO
1900    052A   .   .   .   GDC030 EQU  $           ;YES - PROCESS THE CHARACTER
1901    052A   CD  50  03         CALL CHINT       ;PERFORM INPUT PROCEDURE
1902    052D   CA  0C  05         JZ   GDC010      ;FAST STORE - DO SHORT LOOP
1903    0530   .   .   .   ;
1904    0530   .   .   .   GETDC1 EQU  $           ;SET THE DISPLAY CURSOR
1905    0530   CD  9E  0F         CALL DISLN1      ;SET DISPLAY CURSOR ROW AND
1906    0533   C3  A0  03         JMP  CHINT2        ;COLUMN AND EXIT Z-TRUE
```

```
==============================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                        PAGE  53
==============================================================================
1908    0536    .    .    .   ;*********************************
1909    0536    .    .    .   ; PROCESS DATA COMM INPUT ERROR *
1910    0536    .    .    .   ;*********************************
1911    0536    .    .    .   GDC050 EQU   $
1912    0536    C2   54   12         JNZ   HANG00     ;REPORT AND HANG IF FATAL
1913    0539    CD   47   10         CALL  CKDSPF     ;DISPLAY FUNCTIONS ENABLED?
1914    053C    CC   95   04         CZ    ESCEND     ;NO - FORCE ESC SEQ ABORT
1915    053F    0E   7F   .          MVI   C,ADEL     ;FORCE RUBOUT CHARACTER TO
1916    0541    C3   14   05         JMP   GDC020       ;BE DISPLAYED
1917    0544    .    .    .   ;********************************************
1918    0544    .    .    .   ; NOT IN REMOTE MODE - SET TO LOCAL IF NOT *
1919    0544    .    .    .   ;    IN LOCAL MODE ALREADY                   *
1920    0544    .    .    .   ;********************************************
1921    0544    .    .    .   GDC100 EQU   $
1922    0544    E6   10   .          ANI   REMSET     ;FIRST TIME IN LOCAL?
1923    0546    C4   C7   13         CNZ   ENTLCL     ;YES - SET TO LOCAL MODE
1924    0549    3C   .    .          INR   A          ;FORCE Z FALSE
1925    054A    C9   .    .          RET              ;RETURN NO DATA COMM INPUT
```

```
054B   .   .   .    ;**********************************
054B   .   .   .    ; GTBLK - GET A NEW DISPLAY BLOCK *
054B   .   .   .    ;**********************************
054B   .   .   .    ;
054B   .   .   .    ;   ENTRY:  DON'T CARE
054B   .   .   .    ;
054B   .   .   .    ;   EXIT :  Z - NO BLOCKS AVAILABLE (MEMORY LOCKED)
054B   .   .   .    ;             ALL REGISTERS DESTROYED
054B   .   .   .    ;           NZ - BLOCK ALLOCATED
054B   .   .   .    ;             B,A = H,L = ADDRESS OF CHARACTER
054B   .   .   .    ;                PRECEDING NEXT BLOCK LINK IN BLOCK
054B   .   .   .    ;             C,D,E DESTROYED
054B   .   .   .    ;
054B   .   .   .    GTBLKF EQU  $          ;GET BLOCK FOR SINGLE CHAR I
054B   3E  C3  .           MVI  A,FILL     ;SET FILL CHARACTER TO FILL
054D   .   .   .    ;
054D   .   .   .    GTBLK  EQU  $
054D   32  8F  FF          STA  FILCHR     ;SAVE FILL CHARACTER
0550   2A  AC  FF          LHLD FRBLKS     ;GET POINTER TO FIRST
0553   EB  .   .           XCHG              ;FREE BLOCK IN D,E
0554   7B  .   .           MOV  A,E        ;PUT LSB OF LINK IN A-REG
0555   B7  .   .           ORA  A          ;ANY BLOCKS AVAILABLE?
0556   CC  13  06          CZ   PTBLK      ;NO - RELEASE BLOCKS
0559   CA  07  0B          JZ   MLOCK        ;AND FORCE MEMORY LOCK ON
055C   E6  F0  .           ANI  377Q-BLKSM  ;COMPUTE ADDRESS OF
055E   6F  .   .           MOV  L,A            ;NEXT BLOCK LINK
055F   62  .   .           MOV  H,D
0560   7E  .   .           MOV  A,M        ;GET LSB OF NEXT BLOCK LINK
0561   4F  .   .           MOV  C,A        ;SAVE LSB IN C-REGISTER
0562   2F  .   .           CMA             ;END OF LINE LINK (LOWER
0563   E6  0F  .           ANI  BLKSM        ;FOUR BITS # ALL ONES)?
0565   CA  76  05          JZ   GBL100     ;NO - RELEASE NEXT BLOCK
0568   .   .   .    ;******************************
0568   .   .   .    ; RELEASE LAST BLOCK OF LINE *
0568   .   .   .    ;******************************
0568   13  .   .           INX  D          ;SET H,L TO LSB PART OF PREV
0569   13  .   .           INX  D            ;LINE LINK
056A   6B  .   .           MOV  L,E
056B   CD  6D  19          CALL CHAIN      ;GET PREV LINE ADDR IN H,L
056E   22  AC  FF          SHLD FRBLKS     ;SET AS NEW FREE BLOCKS HEAD
0571   42  .   .           MOV  B,D        ;PUT CURRENT BLOCK ADDRESS
0572   7B  .   .           MOV  A,E          ;IN B,A
0573   C3  84  05          JMP  GBL200     ;FILL BLOCK WITH FILL CHARS
```

```
1971    0576   .    .    .   ;****************************
1972    0576   .    .    .   ; RELEASE NEXT BLOCK OF LINE *
1973    0576   .    .    .   ;****************************
1974    0576   .    .    .   GBL100 EQU  $
1975    0576   23   .    .          INX  H            ;GET MSB OF NEXT BLOCK LINK
1976    0577   46   .    .          MOV  B,M
1977    0578   2B   .    .          DCX  H            ;RESTORE H,L TO ADDRESS OF
1978    0579   .    .    .   ;                          OF LSB PART IN FIRST BLOCK
1979    0579   79   .    .          MOV  A,C          ;COMPUTE ADDRESS OF NEXT
1980    057A   E6   F0   .          ANI  377Q-BLKSM  ;BLOCK LINK IN SECOND BLOC
1981    057C   4F   .    .          MOV  C,A
1982    057D   0A   .    .          LDAX B            ;TRANSFER NEXT BLOCK LINK OF
1983    057E   77   .    .          MOV  M,A            ;SECOND BLOCK TO NEXT BLOC
1984    057F   03   .    .          INX  B              ;LINK IN FIRST BLOCK
1985    0580   23   .    .          INX  H
1986    0581   0A   .    .          LDAX B
1987    0582   77   .    .          MOV  M,A
1988    0583   79   .    .          MOV  A,C          ;SET A-REGISTER FOR "BLNKFL"
1989    0584   .    .    .   ;*******************************************
1990    0584   .    .    .   ; FILL BLOCK WITH SPECIFIED FILL CHARACTER *
1991    0584   .    .    .   ;*******************************************
1992    0584   .    .    .   ;
1993    0584   .    .    .   ;  B,A = ANY ADDRESS IN BLOCK
1994    0584   .    .    .   ;  FILCHR = CHARACTER TO FILL BLOCK WITH
1995    0584   .    .    .   ;
1996    0584   .    .    .   GBL200 EQU  $
1997    0584   F6   0F   .          ORI  BLKSM        ;SET H,L TO ADDRESS OF LAST
1998    0586   6F   .    .          MOV  L,A            ;DISPLAY CHARACTER POSITIO
1999    0587   60   .    .          MOV  H,B            ;IN BLOCK
2000    0588   0E   0D   .          MVI  C,BLKSZ-3  ;SET FILL COUNT
2001    058A   3A   8F   FF         LDA  FILCHR       ;GET THE FILL CHARACTER
2002    058D   .    .    .   GBL210 EQU  $
2003    058D   77   .    .          MOV  M,A          ;STORE THE FILL CHARACTER
2004    058E   2B   .    .          DCX  H            ;MOVE TO NEXT BYTE
2005    058F   0D   .    .          DCR  C            ;BLOCK FILL COMPLETED?
2006    0590   C2   8D   05         JNZ  GBL210       ;NO - CONTINUE FILLING
2007    0593   77   .    .          MOV  M,A          ;YES - WRITE LAST PAD
2008    0594   7D   .    .          MOV  A,L          ;SET B,A TO EXIT ADDRESS
2009    0595   B7   .    .          ORA  A            ;SET NZ
2010    0596   C9   .    .          RET               ;RETURN
```

```
===============================================================================
ITEM   LOC   OBJECT CODE  SOURCE STATEMENTS
===============================================================================
2012   0597   .    .    .   ;****************************
2013   0597   .    .    .   ; GTNWLN - START A NEW LINE *
2014   0597   .    .    .   ;****************************
2015   0597   .    .    .   ;
2016   0597   .    .    .   ;  ENTRY:  LLINE = ADDRESS OF PREVIOUS LINE
2017   0597   .    .    .   ;
2018   0597   .    .    .   ;  EXIT :  NZ - NO BLOCKS AVAILABLE (MEMORY LOCK)
2019   0597   .    .    .   ;             ALL REGISTERS DESTROYED
2020   0597   .    .    .   ;          Z - LINE ALLOCATED
2021   0597   .    .    .   ;             H,L = ADDR OF FIRST CHAR IN NEW LINE
2022   0597   .    .    .   ;             LLINE = ADDR OF LSB PART OF NEXT LINE
2023   0597   .    .    .   ;               POINTER IN THE NEW LINE
2024   0597   .    .    .   ;             A-E DESTROYED
2025   0597   .    .    .   ;
2026   0597   .    .    .   ;  NEW LINE IS LINKED TO PREVIOUS LINE IF PREVIOUS
2027   0597   .    .    .   ;  LINE EXISTS (I.E., LSB OF PREV LINE ADDR # 0)
2028   0597   .    .    .   ;
2029   0597   .    .    .   GTNWLN EQU  $
2030   0597   3E   C0   .          MVI  A,STPR      ;SET LAST FORMAT CONTROL COD
2031   0599   32   C5   FF         STA  LSTFMT        ;TO START PROTECT
2032   059C   CD   4B   05         CALL GTBLKF      ;GET A BLOCK FROM FREE LIST
2033   059F   CA   01   0B         JZ   NZEXIT      ;RETURN NZ IF NO BLOCKS
2034   05A2   EB   .    .          XCHG             ;D,E = NEW BLOCK ADDRESS
2035   05A3   2A   A1   FF         LHLD LLINE       ;GET ADDRESS OF PREVIOUS
2036   05A6   EB   .    .          XCHG               ;LINE IN D,E
2037   05A7   F6   0F   .          ORI  BLKSM       ;COMPUTE ADDRESS OF LSB PART
2038   05A9   D6   02   .          SUI  2             ;OF NEXT LINE LINK
2039   05AB   2B   .    .          DCX  H           ;STORE ADDRESS INTO NEXT
2040   05AC   70   .    .          MOV  M,B           ;BLOCK LINK
2041   05AD   2D   .    .          DCR  L             ;(USE DCR TO AVOID CARRY)
2042   05AE   77   .    .          MOV  M,A
2043   05AF   C6   02   .          ADI  2           ;SET ADDRESS TO MSB PART OF
2044   05B1   6F   .    .          MOV  L,A           ;PREVIOUS LINE LINK
2045   05B2   72   .    .          MOV  M,D         ;SET PREVIOUS LINE LINK TO
2046   05B3   2B   .    .          DCX  H             ;POINT TO OLD LINE
2047   05B4   73   .    .          MOV  M,E
2048   05B5   2B   .    .          DCX  H
2049   05B6   36   CE   .          MVI  M,EOP       ;SET NEXT LINE LINK TO "EOP"
2050   05B8   2B   .    .          DCX  H
2051   05B9   AF   .    .          XRA  A           ;SET TERMINATOR (LSB = 0)
2052   05BA   77   .    .          MOV  M,A
2053   05BB   22   A1   FF         SHLD LLINE       ;STORE NEW LAST LINE ADDRESS
2054   05BE   2B   .    .          DCX  H
2055   05BF   CD   68   0D         CALL STCHR1      ;SET FIRST DISPLAY CHARACTER
```

```
 2057    05C2    .   .   .   ;*******************************************
 2058    05C2    .   .   .   ; LINK NEW LINE BACK TO PREVIOUS LAST LINE *
 2059    05C2    .   .   .   ;*******************************************
 2060    05C2    B3  .   .          ORA   E          ;PREVIOUS LINE EXIST (LSB#0)
 2061    05C3    C8  .   .          RZ               ;NO - RETURN
 2062    05C4    EB  .   .          XCHG             ;YES - LINK NEW LINE TO
 2063    05C5    73  .   .          MOV   M,E          ;PREVIOUS LINE
 2064    05C6    23  .   .          INX   H
 2065    05C7    72  .   .          MOV   M,D
 2066    05C8    EB  .   .          XCHG             ;RESTORE H,L
 2067    05C9    BF  .   .          CMP   A          ;SET Z TRUE
 2068    05CA    C9  .   .          RET              ;RETURN
```

```
05CB  .  .  .  ;****************************************
05CB  .  .  .  ; INITDS - SET UP INITIAL DISPLAY VALUES *
05CB  .  .  .  ;****************************************
05CB  .  .  .  ;
05CB  .  .  .  ;  EXIT :  H,L = ADDRESS OF THE LSB PART OF THE
05CB  .  .  .  ;               NEXT LINE POINTER IN THE INITIAL
05CB  .  .  .  ;               DISPLAY BLOCK
05CB  .  .  .  ;            A DESTROYED
05CB  .  .  .  ;
05CB  .  .  .  ;  THIS ROUTINE ALLOCATES THE INITIAL LINE OF
05CB  .  .  .  ;  THE DISPLAY AND INITIALIZES THE DISPLAY
05CB  .  .  .  ;  PARAMETERS:
05CB  .  .  .  ;
05CB  .  .  .  ;    DISPST,CURADR =  ADDRESS OF THE FIRST DISPLAY
05CB  .  .  .  ;      CHARACTER IN THE INITIAL DISPLAY BLOCK
05CB  .  .  .  ;
05CB  .  .  .  ;    LSTLIN,FLINE,TOPLIN =  ADDRESS OF THE LSB
05CB  .  .  .  ;      PART OF THE NEXT LINE POINTER IN THE
05CB  .  .  .  ;      INITIAL DISPLAY BLOCK
05CB  .  .  .  ;
05CB  .  .  .  ;    RHTMGN = MAXCOL (= 79)
05CB  .  .  .  ;
05CB  .  .  .  INITDS EQU  $
05CB  CD 97 05        CALL GTNWLN     ;GET INITIAL DISPLAY BLOCK
05CE  22 FE FF        SHLD DISPST     ;SET THE DISPLAY POINTER
05D1  22 C3 FF        SHLD CURADR       ;AND THE CURRENT CHAR ADDR
05D4  23 .  .         INX  H
05D5  22 C9 FF        SHLD LSTLIN     ;SET THE CURRENT LINE
05D8  22 9F FF        SHLD FLINE        ;PARAMETERS
05DB  22 CB FF        SHLD TOPLIN
05DE  3E 4F .         MVI  A,MAXCOL   ;INITIALIZE THE RIGHT MARGIN
05E0  32 BE FF        STA  RHTMGN       ;TO THE LAST COLUMN
05E3  C9 .  .         RET             ;RETURN
```

```
2104   05E4   .    .    .   ;***********************************
2105   05E4   .    .    .   ; LOCLIN - PROCESS LOCAL DATA ENTRY *
2106   05E4   .    .    .   ;***********************************
2107   05E4   .    .    .   ;
2108   05E4   .    .    .   ;  ENTRY:  C = INPUT CHARACTER
2109   05E4   .    .    .   ;          (CHARIN) = KEYBOARD INPUT CODE
2110   05E4   .    .    .   ;
2111   05E4   .    .    .   ;  EXIT :  ALL REGISTERS DESTROYED
2112   05E4   .    .    .   ;
2113   05E4   .    .    .   ;  THIS ROUTINE PROCESSES INPUT CHARACTERS FROM
2114   05E4   .    .    .   ;  KEYBOARD.  THE ROUTINE DETERMINES WHETHER OR
2115   05E4   .    .    .   ;  NOT THE CHARACTER SHOULD BE TRANSMITTED OR
2116   05E4   .    .    .   ;  PROCESSED LOCALLY
2117   05E4   .    .    .   ;
2118   05E4   .    .    .   ;  LOCL10 - PROCESS FUNCTIONAL KEY INPUT
2119   05E4   .    .    .   ;
2120   05E4   .    .    .   LOCL10 EQU  $
2121   05E4   3A   FB   FF         LDA  KBJMPR     ;GET KEYBOARD JUMPERS A-H
2122   05E7   E6   01   .          ANI  CONDIS     ;DISPLAY ALL FUNCTIONS OR
2123   05E9   CC   47   10         CZ   CKDSPF       ;DISPLAY FUNCTIONS ENABLED
2124   05EC   CA   0A   06         JZ   LC1050     ;NO - PROCESS LOCALLY ONLY
2125   05EF   .    .    .   ;*******************************************
2126   05EF   .    .    .   ; TRANSMIT CODE IF IN REMOTE CHARACTER MODE *
2127   05EF   .    .    .   ;*******************************************
2128   05EF   .    .    .   LOCLIN EQU  $
2129   05EF   CD   8C   19         CALL CHKSFK     ;SOFT KEY DEFINE MODE?
2130   05F2   C2   0A   06         JNZ  LC1050     ;YES - PROCESS LOCALLY ONLY
2131   05F5   3A   F3   FF         LDA  MDFLG2     ;NO - GET HARD MODE FLAGS
2132   05F8   E6   0A   .          ANI  REMOTE+BLKMDE
2133   05FA   EE   08   .          XRI  REMOTE     ;REMOTE AND NOT BLOCK MODE?
2134   05FC   C2   0A   06         JNZ  LC1050     ;NO - PROCESS LOCALLY ONLY
2135   05FF   79   .    .          MOV  A,C        ;YES - RECALL THE INPUT
2136   0600   CD   C1   17         CALL XPUTDC     ;OUTPUT THE CHARACTER
2137   0603   D8   .    .          RC                ;(RETURN IF OUTPUT ERROR)
2138   0604   3A   FC   FF         LDA  KBDCSW     ;GET THE DATA COMM SWITCHES
2139   0607   E6   80   .          ANI  FULDUP     ;FULL DUPLEX?
2140   0609   C0   .    .          RNZ             ;YES - RETURN
2141   060A   .    .    .   ;                       NO - PROCESS INPUT LOCALLY
2142   060A   .    .    .   ;***************************
2143   060A   .    .    .   ; PROCESS THE INPUT LOCALLY *
2144   060A   .    .    .   ;***************************
2145   060A   .    .    .   ;
2146   060A   .    .    .   ;  INPUT CHARACTER IN C-REGISTER
2147   060A   .    .    .   ;
2148   060A   .    .    .   LC1050 EQU  $
2149   060A   CD   76   19         CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
2150   060D   C2   B9   03         JNZ  CHINT1     ;YES - FORCE FULL PROCESSING
2151   0610   C3   50   03         JMP  CHINT      ;NO - 1ST TRY FAST PROCESSIN
```

```
2153    0613   .    .    .    ;********************************************
2154    0613   .    .    .    ; PTBLK - RELEASE A LINE TO THE FREE LIST FROM *
2155    0613   .    .    .    ;   THE DISPLAY LIST                          *
2156    0613   .    .    .    ;********************************************
2157    0613   .    .    .    ;
2158    0613   .    .    .    ;  ENTRY:  DON'T CARE
2159    0613   .    .    .    ;
2160    0613   .    .    .    ;  EXIT :  Z - LINE NOT RELEASED
2161    0613   .    .    .    ;            NC - MEMORY LOCKED
2162    0613   .    .    .    ;            C - OUTPUT FAILED FOR EDIT MODE
2163    0613   .    .    .    ;            ALL REGISTERS DESTROYED
2164    0613   .    .    .    ;          NZ - LINE RELEASED
2165    0613   .    .    .    ;            D,E = ADDRESS OF FIFTH BYTE FROM
2166    0613   .    .    .    ;            A = E
2167    0613   .    .    .    ;            B,C,H,L DESTROYED
2168    0613   .    .    .    ;
2169    0613   .    .    .    PTBLK   EQU   $
2170    0613   CD   81   19           CALL CHKMLK     ;MEMORY LOCK ENABLED?
2171    0616   CA   07   0B           JZ    MLOCK     ;YES - FLAG MEMORY FULL
2172    0619   CD   47   06           CALL PTB100     ;SWITCH DISPLAY PARAMETERS
2173    061C   .    .    .    ;                         IF IN SOFT KEY MODE
2174    061C   3A   9A   FF           LDA   NROWS     ;GET NUMBER OF ROWS NEEDED
2175    061F   B7   .    .            ORA   A         ;NEW ROWS BEING ADDED?
2176    0620   CC   4D   10           CZ    CKEDIT    ;NO - EDIT MODE?
2177    0623   C2   51   06           JNZ   PTB200    ;YES - RELEASE TOP LINE
2178    0626   2A   C9   FF           LHLD  LSTLIN    ;NO - GET CURRENT LINE ADDR
2179    0629   B6   .    .            ORA   M         ;CURRENTLY IN THE LAST LINE
2180    062A   CA   51   06           JZ    PTB200    ;YES - RELEASE TOP LINE
2181    062D   .    .    .    ;                        NO - RELEASE BOTTOM LINE
```

```
==============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                          PAGE  61
==============================================================================
 2183    062D   .   .   .   ;*****************************
 2184    062D   .   .   .   ; RELEASE LAST LINE OF MEMORY *
 2185    062D   .   .   .   ; UPDATE LAST LINE POINTER    *
 2186    062D   .   .   .   ;*****************************
 2187    062D   2A  A1  FF        LHLD LLINE      ;GET LAST LINE ADDRESS
 2188    0630   23  .   .         INX  H          ;GET PREVIOUS LINE ADDRESS
 2189    0631   23  .   .         INX  H
 2190    0632   5E  .   .         MOV  E,M
 2191    0633   23  .   .         INX  H
 2192    0634   56  .   .         MOV D,M
 2193    0635   EB  .   .         XCHG
 2194    0636   22  A1  FF        SHLD LLINE      ;SET PREV LINE AS LAST LINE
 2195    0639   .   .   .   ;****************************
 2196    0639   .   .   .   ; STORE EOP IN NEW LAST LINE *
 2197    0639   .   .   .   ;****************************
 2198    0639   36  00  .         MVI  M,0        ;SET TERMINATOR CODE IN
 2199    063B   23  .   .         INX  H            ;NEW LAST LINE
 2200    063C   36  CE  .         MVI  M,EOP
 2201    063E   1B  .   .         DCX  D          ;SET D,E TO POINT TO LSB PAR
 2202    063F   1B  .   .         DCX  D            ;NEXT LINE POINTER IN OLD
 2203    0640   1B  .   .         DCX  D            ;LAST LINE
 2204    0641   C3  8C  06        JMP  PTB300     ;ADD LINE TO FREE LIST
 2205    0644   .   .   .   ;*****************************************
 2206    0644   .   .   .   ; PTB100 - SET PROPER DISPLAY PARAMETERS *
 2207    0644   .   .   .   ;*****************************************
 2208    0644   .   .   .   PTB090 EQU  $           ;I/O OUTPUT FAIL EXIT
 2209    0644   CD  19  0B        CALL MLK010     ;CLEAR ROWS ALLOCATED FLAG
 2210    0647   .   .   .   PTB100 EQU  $
 2211    0647   CD  8C  19        CALL CHKSFK     ;SOFT KEY DEFINE MODE?
 2212    064A   37  .   .         STC               ;(SET C-FLAG FOR I/O FAIL)
 2213    064B   C2  6F  21        JNZ  SWAP1      ;YES - SWAP DISPLAY PARMS
 2214    064E   C9  .   .         RET             ;NO - RETURN
 2215    064F   00  .   .         NOP             ;"NOP'S" FOR PATCH TO "PT772
 2216    0650   00  .   .         NOP
 2218   0651   .    .    .    ;********************************
 2219   0651   .    .    .    ; RELEASE FIRST LINE OF MEMORY *
 2220   0651   .    .    .    ;********************************
 2221   0651   .    .    .    PTB200 EQU $
 2222   0651   2A   CB   FF          LHLD TOPLIN     ;GET TOP LINE ADDRESS
 2223   0654   23   .    .           INX  H          ;SET FOR PREVIOUS LINE
 2224   0655   23   .    .           INX  H             ;ADDRESS
 2225   0656   7E   .    .           MOV A,M
 2226   0657   B7   .    .           ORA A         ;TOP LINE = FIRST LINE?
 2227   0658   C2   71   06          JNZ PTB220    ;FIRST LINE IS NOT TOP LINE
 2228   065B   .    .    .    ;**********************************************
 2229   065B   .    .    .    ; TOP LINE OF DISPLAY IS FIRST LINE OF MEMORY *
 2230   065B   .    .    .    ; DO ROLL-UP                                  *
 2231   065B   .    .    .    ;**********************************************
 2232   065B   21   C0   FF          LXI  H,CURROW
 2233   065E   3A   6B   FF          LDA  MLKROW     ;USER WORKING IN FIRST
 2234   0661   BE   .    .           CMP  M             ;UNLOCKED ROW?
 2235   0662   C4   27   0C          CNZ  ROLLUP     ;NO - ROLL UP DISPLAY
 2236   0665   CA   04   0B          JZ   MLOCK0     ;ROLL UP FAIL - LOCK MEMORY
 2237   0668   21   C0   FF          LXI  H,CURROW   ;DECREMENT CURSOR ROW
 2238   066B   46   .    .           MOV  B,M
 2239   066C   05   .    .           DCR B
 2240   066D   FA   71   06          JM   PTB220   ;DON'T STORE IF ROW = 0
 2241   0670   70   .    .           MOV M,B
 2242   0671   .    .    .    ;*****************************
 2243   0671   .    .    .    ; ADVANCE FIRST LINE POINTER *
 2244   0671   .    .    .    ;*****************************
 2245   0671   .    .    .    PTB220 EQU  $
 2246   0671   2A   9F   FF          LHLD FLINE      ;GET ADDRESS OF FIRST DISPLA
 2247   0674   EB   .    .           XCHG               ;LINE
 2248   0675   CD   4D   10          CALL CKEDIT     ;EDIT MODE ENABLED?
 2249   0678   C4   32   28          CNZ  PTTPLN     ;YES - TRY TO OUTPUT LINE
 2250   067B   DA   44   06          JC   PTB090     ;OUTPUT FAILED - RETURN FAIL
 2251   067E   EB   .    .           XCHG            ;PUT ADDRESS BACK INTO D,E
 2252   067F   5E   .    .           MOV  E,M        ;GET ADDRESS OF NEW FIRST
 2253   0680   23   .    .           INX  H            ;FIRST LINE
 2254   0681   56   .    .           MOV  D,M
 2255   0682   13   .    .           INX  D          ;SET TO NEXT LINE POINTER
 2256   0683   EB   .    .           XCHG
 2257   0684   22   9F   FF          SHLD  FLINE     ;STORE AS NEW FIRST LINE
 2258   0687   .    .    .    ;********************************************
 2259   0687   .    .    .    ; CLEAR PREVIOUS LINE PNTR IN NEW FIRST LINE *
 2260   0687   .    .    .    ;********************************************
 2261   0687   23   .    .           INX  H          ;ADVANCE TO PREVIOUS LINE
 2262   0688   23   .    .           INX  H            ;POINTER
 2263   0689   36   00   .           MVI  M,0        ;ZERO LSB TO FLAG AS TOP LIN
 2264   068B   1B   .    .           DCX  D          ;SET D,E TO LSB OF NEXT LINE
 2265   068C   .    .    .    ;                        POINTER IN LINE TO BE
 2266   068C   .    .    .    ;                        RELEASED
 2268   068C    .    .    .  ;******************************
 2269   068C    .    .    .  ; RELEASE LINE                 *
 2270   068C    .    .    .  ; D,E = START ADDRESS OF LINE *
 2271   068C    .    .    .  ;******************************
 2272   068C    .    .    .  PTB300 EQU  $
 2273   068C   D5    .    .         PUSH D          ;SAVE REGISTERS D,E
 2274   068D   CD   47   06         CALL PTB100     ;RESTORE PROPER DISPLAY PARM
 2275   0690   D1    .    .         POP  D          ;RESTORE D,E
 2277    0691   .    .    .   ;*********************************
 2278    0691   .    .    .   ; PUTLIN - ADD LINE TO FREE LIST *
 2279    0691   .    .    .   ;*********************************
 2280    0691   .    .    .   ;
 2281    0691   .    .    .   ;   ENTRY:  D,E = ADDRESS OF NEXT LINE FIELD'S LSB
 2282    0691   .    .    .   ;                OF LINE TO BE RELEASED
 2283    0691   .    .    .   ;
 2284    0691   .    .    .   ;   EXIT :  D,E UNCHANGED
 2285    0691   .    .    .   ;           A = E
 2286    0691   .    .    .   ;           Z FALSE
 2287    0691   .    .    .   ;           H,L DESTROYED
 2288    0691   .    .    .   ;           FREE BLOCKS LIST UPDATED TO INCLUDE
 2289    0691   .    .    .   ;             RELEASE LINE
 2290    0691   .    .    .   ;
 2291    0691   .    .    .   PUTLIN EQU  $
 2292    0691   CD   C0   10         CALL MLKOF       ;RESET MEMORY LOCKED FLAG
 2293    0694   2A   AC   FF         LHLD FRBLKS      ;GET CURRENT FREE BLOCKS HEA
 2294    0697   EB   .    .          XCHG             ;SET H,L TO MSB PART OF NEXT
 2295    0698   22   AC   FF         SHLD FRBLKS      ;SET FREE BLOCKS POINTER TO
 2296    069B   7D   .    .          MOV  A,L           ;RELEASED LINE
 2297    069C   23   .    .          INX  H           ;PUT PREVIOUS FREE BLOCKS
 2298    069D   23   .    .          INX  H             ;HEAD INTO PREVIOUS LINE
 2299    069E   73   .    .          MOV  M,E           ;POINTER OF RELEASED LINE
 2300    069F   2C   .    .          INR  L             ;(USE INR TO FORCE NZ)
 2301    06A0   72   .    .          MOV  M,D
 2302    06A1   EB   .    .          XCHG             ;RELEASED LINE ADDRESS IN D,
 2303    06A2   5F   .    .          MOV  E,A         ;SET A = E
 2304    06A3   C9   .    .          RET              ;RETURN
```

```
=======================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                  PAGE  65
=======================================================================
2306    06A4    .    .    .    ;*******************************************
2307    06A4    .    .    .    ; RCADRA - LOCATE CURRENT CURSOR POSITION  . *
2308    06A4    .    .    .    ;   IF POSITION EXIST - DON'T EXTEND DISPLAY *
2309    06A4    .    .    .    ;*******************************************
2310    06A4    .    .    .    ;
2311    06A4    .    .    .    ;  ENTRY:  DON'T CARE
2312    06A4    .    .    .    ;
2313    06A4    .    .    .    ;  EXIT :  SEE "RCADDR"
2314    06A4    .    .    .    ;
2315    06A4    .    .    .    RCADRA EQU   $
2316    06A4    CD   23   20          CALL  CRADV1     ;CLEAR CURSOR ADVANCE FLAG
2317    06A7    3E   01   .           MVI   A,IGNTRM   ;SET TO IGNORE NON-DISPLAYIN
2318    06A9    32   6D   FF          STA   TRMFCT       ;TERMINATOR
2319    06AC    .    .    .    RCADRB EQU   $
2320    06AC    3E   FF   .           MVI   A,377Q     ;SET "BLKFIL" TO INHIBIT
2321    06AE    32   91   FF          STA   BLKFIL       ;LINE EXTENSION
2322    06B1    C3   B8   06          JMP   RCADR2     ;LOCATE CURSOR POSITION
2323    06B4    .    .    .    ;****************************************
2324    06B4    .    .    .    ; LOCATE ADDR CORRESPONDING TO ROW/COLUMN *
2325    06B4    .    .    .    ; DO NOT ADD ROWS IF ROW DOES NOT EXIST   *
2326    06B4    .    .    .    ;****************************************
2327    06B4    .    .    .    RCADR1 EQU   $
2328    06B4    AF   .    .           XRA   A          ;SET TO LOCATE COLUMN 0
2329    06B5    32   C1   FF          STA   CURCOL       ;IN DESIRED ROW
2330    06B8    .    .    .    RCADR2 EQU   $
2331    06B8    3A   C1   FF          LDA   CURCOL     ;GET THE CURRENT COLUMN
2332    06BB    .    .    .    RCADR3 EQU   $
2333    06BB    21   9A   FF          LXI   H,NROWS    ;SET "NROWS" TO INHIBIT
2334    06BE    36   FF   .           MVI   M,377Q       ;BUILDING OF NEW ROWS
2335    06C0    CD   0B   07          CALL  RCADR0     ;FIND CHARACTER ADDRESS
2336    06C3    21   9A   FF          LXI   H,NROWS    ;RESET BUILD INHIBIT FLAGS
2337    06C6    36   00   .           MVI   M,0          ;WITHOUT CHANGING PROCESSO
2338    06C8    2E   91   .           MVI   L,BLKFIL-BASE  ;FLAGS
2339    06CA    36   00   .           MVI   M,0
2340    06CC    C9   .    .           RET              ;RETURN
```

```
 2342   06CD    .    .    .    ;********************************************
 2343   06CD    .    .    .    ; RCADR4 - GET ADDRESS OF FIRST CHARACTER AFTER *
 2344   06CD    .    .    .    ;   AFTER PREVIOUS ROW AND COLUMN              *
 2345   06CD    .    .    .    ;********************************************
 2346   06CD    .    .    .    ;
 2347   06CD    .    .    .    ;          ENTRY:  CURROW = CURRENT ROW
 2348   06CD    .    .    .    ;                  CURCOL = CURRENT COLUMNN
 2349   06CD    .    .    .    ;
 2350   06CD    .    .    .    ;          EXIT :  Z - CHARACTER FOUND
 2351   06CD    .    .    .    ;                    C = COLUMN NUMBER
 2352   06CD    .    .    .    ;                    D,E = CHARACTER ADDRESS
 2353   06CD    .    .    .    ;                    IF FORMAT MODE ENABLED
 2354   06CD    .    .    .    ;                      B = -1, CHARACTER PROTECTED
 2355   06CD    .    .    .    ;                        # -1, CHARACTER NOT PROTECTED
 2356   06CD    .    .    .    ;                    OTHERWISE, B DESTROYED
 2357   06CD    .    .    .    ;                  A,H,L DESTROYED
 2358   06CD    .    .    .    ;                  NZ - CHARACTER NOT FOUND
 2359   06CD    .    .    .    ;                    ALL REGISTERS DESTROYED
 2360   06CD    .    .    .    ;
 2361   06CD    .    .    .    RCADR4 EQU $
 2362   06CD   3A   C1   FF           LDA  CURCOL     ;GET CURRENT COLUMN NUMBER
 2363   06D0   3D    .    .           DCR  A          ;SET FOR PREVIOUS COLUMN
 2364   06D1   CD   BB   06           CALL RCADR3     ;DOES CHARACTER EXIST
 2365   06D4   C0    .    .           RNZ             ;NO - RETURN
 2366   06D5   4F    .    .           MOV  C,A        ;YES - SAVE COLUMN FOUND IN
 2367   06D6   0C    .    .           INR  C          ;ADVANCE TO NEXT COLUMN
 2368   06D7   CD   87   0B           CALL NXTCHR     ;GET NEXT CHARACTER
 2369   06DA   CD   76   19           CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
 2370   06DD   47    .    .           MOV  B,A          ;(SET B TO INDICATE NOT
 2371   06DE    .    .    .    ;                            PROTECTED IF NOT FORMAT)
 2372   06DE    .    .    .    ;                            NEXT STATEMENT RETURNS)
 2373   06DE   C8    .    .           RZ             ;NO - RETURN
 2375    06DF   .   .   .   ;
 2376    06DF   .   .   .   ;  FORMAT MODE - SEE IF NEXT ASCII CHAR PROTECTED
 2377    06DF   .   .   .   ;
 2378    06DF   CD  65  10         CALL CKPROT     ;PREVIOUS CHAR PROTECTED?
 2379    06E2   CA  F7  06         JZ   RCA440     ;YES - SEE IF NEXT CHAR UNPR
 2380    06E5   .   .   .   ;********************************
 2381    06E5   .   .   .   ; LAST CHAR WAS UNPROTECTED     *
 2382    06E5   .   .   .   ; SEE IF NEXT CHAR IS PROTECTED *
 2383    06E5   .   .   .   ;********************************
 2384    06E5   CD  9D  1E         CALL FNDCHO     ;IS NEXT CHARACTER PROTECTED
 2385    06E8   C2  F7  06         JNZ  RCA440     ;YES - SEE IF NEXT IS UNPROT
 2386    06EB   2A  C3  FF         LHLD CURADR     ;NO - RECALL CURRENT CHAR
 2387    06EE   EB  .   .          XCHG               ;ADDRESS AND PUT INTO D,E
 2388    06EF   .   .   .   RC4010 EQU  $
 2389    06EF   CD  87  0B         CALL NXTCHR     ;GET NEXT DISPLAY CHARACTER
 2390    06F2   06  00  .          MVI  B,0        ;SET B FOR NOT PROTECTED
 2391    06F4   C3  02  07         JMP  RCA460     ;EXIT CHARACTER FOUND
 2392    06F7   .   .   .   ;***********************************
 2393    06F7   .   .   .   ; PROTECT CHAR FOUND               *
 2394    06F7   .   .   .   ; SEE IF SUBSEQUENT UNPROTECT CHAR *
 2395    06F7   .   .   .   ;***********************************
 2396    06F7   .   .   .   RCA440 EQU $
 2397    06F7   21  C2  C1         LXI  H,ENDPR*256+XMONLY  ;IS NEXT CHARACTER
 2398    06FA   CD  A0  1E         CALL FNDCH         ;AN UNPROTECT OR XMIT ONLY
 2399    06FD   C2  EF  06         JNZ  RC4010     ;YES - RETURN UNPROTECTED
 2400    0700   06  FF  .          MVI  B,-1       ;NO - RETURN CHAR PROTECTED
 2401    0702   .   .   .   RCA460 EQU $
 2402    0702   21  C1  FF         LXI  H,CURCOL
 2403    0705   4E  .   .          MOV  C,M        ;RECALL CURSOR COLUMN
 2404    0706   .   .   .   ;
 2405    0706   .   .   .   ;  ZRETRN - RETURN WITH Z-FLAG TRUE
 2406    0706   .   .   .   ;
 2407    0706   .   .   .   ZRETRN EQU  $
 2408    0706   AF  .   .          XRA A         ;SET ZERO FLAG
 2409    0707   C9  .   .          RET           ;RETURN
 2411   0708   .   .   .   ;*************************************************
 2412   0708   .   .   .   ; RCADDR - DETERMINE LOCATION OF ASCII CHARACTER *
 2413   0708   .   .   .   ;   AT SPECIFIED ROW AND COLUMN OF DISPLAY LIST  *
 2414   0708   .   .   .   ;*************************************************
 2415   0708   .   .   .   ;
 2416   0708   .   .   .   ;  ENTRY:  CURROW,CURCOL = DESIRED ROW/COLUMN
 2417   0708   .   .   .   ;          LSTROW,LSTCOL = LAST ROW/COLUMN DONE
 2418   0708   .   .   .   ;          CURADR = ADDRESS CORRESPONDING TO
 2419   0708   .   .   .   ;            LSTROW, LSTCOL
 2420   0708   .   .   .   ;          LSTLIN = ADDRESS OF LINE CORRESPONDING
 2421   0708   .   .   .   ;            TO LSTROW
 2422   0708   .   .   .   ;          NROWS = 0, BUILD NEW ROWS AS NEEDED
 2423   0708   .   .   .   ;                # 0, DON'T BUILD NEW ROWS
 2424   0708   .   .   .   ;          BLKFIL = 0, EXTEND LINE AS NEEDED
 2425   0708   .   .   .   ;                 # 0, DON'T EXTEND LINE
 2426   0708   .   .   .   ;
 2427   0708   .   .   .   ;  EXIT :  Z - CHARACTER FOUND
 2428   0708   .   .   .   ;            A,B,C,L DESTROYED
 2429   0708   .   .   .   ;          NZ - CHARACTER NOT FOUND
 2430   0708   .   .   .   ;            M - ROWS NOT BUILT
 2431   0708   .   .   .   ;              E = NUMBER OF ROWS NEEDED
 2432   0708   .   .   .   ;            P - ROW LOCATED
 2433   0708   .   .   .   ;              A = COLUMN NUMBER FOUND
 2434   0708   .   .   .   ;              B = ROW NUMBER FOUND
 2435   0708   .   .   .   ;              C = NUMBER OF CHARACTERS NEEDED
 2436   0708   .   .   .   ;          D,E = ADDRESS OF LAST CHARACTER FOUND
 2437   0708   .   .   .   ;          H = BASEH
 2438   0708   .   .   .   ;
 2439   0708   .   .   .   ;          LSTROW,LSTCOL,LSTLIN,CURADR ARE UPDATED
 2440   0708   .   .   .   ;          TO THE LAST CHARACTER FOUND.
 2441   0708   .   .   .   ;
 2442   0708   .   .   .   RCADDR EQU  $
 2443   0708  3A  C1  FF          LDA  CURCOL     ;GET DESIRED COLUMN NUMBER
 2444   070B   .   .   .   RCADR0 EQU  $
 2445   070B  32  85  FF          STA  TMPCOL     ;SAVE DESIRED COLUMN NUMBER
 2446   070E  3A  C0  FF          LDA  CURROW     ;GET THE DESIRED ROW NUMBER
 2447   0711  2A  C7  FF          LHLD LSTROW     ;GET LAST ROW AND COLUMN DON
 2448   0714  44   .   .          MOV  B,H        ;PUT LAST COLUMN IN B-REG
 2449   0715  95   .   .          SUB  L          ;MOVED TO A NEW ROW?
 2450   0716  2A  C9  FF          LHLD LSTLIN       ;(GET LAST LINE DONE ADDR)
 2451   0719  CA  5E  07          JZ   RCA240     ;YES - LOCATE COLUMN
 2452   071C   .   .   .   ;**************************
 2453   071C   .   .   .   ; ROW HAS CHANGED        *
 2454   071C   .   .   .   ; LOCATE START OF NEW ROW *
 2455   071C   .   .   .   ;**************************
 2456   071C  5F   .   .          MOV E,A       ;SAVE COUNT
 2457   071D  B7   .   .          ORA A         ;SET FLAGS
 2458   071E  F2  35  07          JP  RCA140    ;ROW IS AHEAD OF THIS ROW
```

```
2460   0721   .    .    .   ;****************************
2461   0721   .    .    .   ; ROW IS BEFORE CURRENT ROW *
2462   0721   .    .    .   ; SEARCH BACK               *
2463   0721   .    .    .   ;****************************
2464   0721   .    .    .   RCA120 EQU $
2465   0721   23   .    .         INX  H          ;SET ADDRESS TO PREVIOUS
2466   0722   23   .    .         INX  H            ;LINE POINTER
2467   0723   CD   6D   19        CALL CHAIN   ;GET ADDRESS OF PREVIOUS ROW
2468   0726   1C   .    .         INR  E          ;ROW FOUND?
2469   0727   C2   21   07        JNZ  RCA120     ;NO - CONTINUE BACKING UP
2470   072A   C3   54   07        JMP  RCA220     ;YES - SET NEW ROW
2471   072D   .    .    .   ;******************************
2472   072D   .    .    .   ; ROW IS AHEAD OF CURRENT ROW *
2473   072D   .    .    .   ; SEARCH AHEAD                *
2474   072D   .    .    .   ;******************************
2475   072D   .    .    .   RCA130 EQU  $
2476   072D   CD   6D   19        CALL CHAIN      ;GET ADDRESS OF NEXT ROW
2477   0730   23   .    .         INX  H          ;SET TO NEXT LINE PTR ADDRES
2478   0731   1D   .    .         DCR  E          ;ROW FOUND?
2479   0732   CA   54   07        JZ   RCA220     ;YES - LOCATE COLUMN
2480   0735   .    .    .   RCA140 EQU  $         ;NO - CHECK FOR ANOTHER ROW
2481   0735   7E   .    .         MOV  A,M        ;GET LSB OF NEXT ROW POINTER
2482   0736   B7   .    .         ORA  A          ;DOES NEXT ROW EXIST?
2483   0737   C2   2D   07        JNZ  RCA130     ;YES - CHECK FOR ROW FOUND
2484   073A   .    .    .   ;*******************
2485   073A   .    .    .   ; ROW NOT IN MEMORY *
2486   073A   .    .    .   ; CREATE NEW ROW    *
2487   073A   .    .    .   ;*******************
2488   073A   .    .    .   RCA200 EQU $
2489   073A   CD   76   19        CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
2490   073D   C2   01   0B        JNZ  NZEXIT     ;YES - DO NOT BUILD ROWS
2491   0740   21   9A   FF        LXI  H,NROWS    ;NO - GET BUILD FLAG
2492   0743   B6   .    .         ORA  M          ;INHIBIT ROW BUILD?
2493   0744   C0   .    .         RNZ             ;YES - RETURN (A = 377B)
2494   0745   73   .    .         MOV  M,E        ;NO - STORE # OF ROWS NEEDED
```

```
LOC   OBJECT CODE  SOURCE STATEMENTS
0746  .   .   .    ;******************************
0746  .   .   .    ; GET NEW ROW AND LINK TO OLD *
0746  .   .   .    ;******************************
0746  .   .   .    RCA210 EQU  $
0746  CD  97  05          CALL GTNWLN     ;ADD A LINE TO THE DISPLAY
0749  C0  .   .           RNZ             ;RETURN FAIL IF MEMORY LOCKE
074A  21  9A  FF          LXI  H,NROWS    ;DECREMENT # OF ROWS NEEDED
074D  35  .   .           DCR  M          ;ALL NEEDED ROWS ALLOCATED?
074E  C2  46  07          JNZ  RCA210     ;NO - GET ANOTHER ROW
0751  .   .   .    ;************************************
0751  .   .   .    ; ALL REQUIRED ROWS HAVE BEEN ADDED *
0751  .   .   .    ;************************************
0751  2A  A1  FF          LHLD LLINE      ;GET START ADDRESS OF ROW
0754  .   .   .    RCA220 EQU  $          ;UPDATE LOCATE COLUMN
0754  CD  9C  0A          CALL LSTLUP     ;SET "LSTLIN" TO NEW ROW
0757  3A  85  FF          LDA  TMPCOL     ;RECALL COLUMN TO BE FOUND
075A  4F  .   .           MOV  C,A        ;PUT COLUMN NUMBER INTO C-RE
075B  C3  69  07          JMP  RCA245     ;GO LOCATE THE COLUMN
```

```
=======================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=======================================================================
2515    075E   .   .   .   ;****************************
2516    075E   .   .   .   ; CURRENT ROW = DESIRED ROW *
2517    075E   .   .   .   ; CHECK COLUMN              *
2518    075E   .   .   .   ;****************************
2519    075E   .   .   .   RCA240 EQU  $
2520    075E   3A  85  FF         LDA  TMPCOL    ;GET THE DESIRED COLUMN
2521    0761   4F  .   .          MOV  C,A       ;PUT IT INTO THE C-REGISTER
2522    0762   90  .   .          SUB  B         ;COLUMN WANTED >= LAST DONE?
2523    0763   F2  71  07         JP   RCA250    ;YES - SCAN FORWARD
2524    0766   .   .   .   ;*******************************************
2525    0766   .   .   .   ; DESIRED COLUMN LESS THAN CURRENT COLUMN *
2526    0766   .   .   .   ; START SEARCH AT BEGINNING OF ROW        *
2527    0766   .   .   .   ;*******************************************
2528    0766   CD  9F  0A         CALL LSTLU1    ;SET LINE START PARAMETERS
2529    0769   .   .   .   ;                      (PUTS H,L INTO D,E)
2530    0769   .   .   .   RCA245 EQU  $
2531    0769   3E  01  .          MVI  A,IGNTRM  ;SET FUNCTION FLAG TO IGNORE
2532    076B   32  6D  FF         STA  TRMFCT      ;NON-DISPLAYING TERMINATOR
2533    076E   C3  7B  07         JMP RCA255   ;GO LOCATE COLUMN
2534    0771   .   .   .   ;*********************************************
2535    0771   .   .   .   ; DESIRED COLUMN AT OR PAST CURRENT COLUMN *
2536    0771   .   .   .   ; START SEARCH AT CURRENT COLUMN           *
2537    0771   .   .   .   ;*********************************************
2538    0771   .   .   .   RCA250 EQU $
2539    0771   4F  .   .          MOV  C,A        ;SAVE # OF COLUMNS TO ADVANC
2540    0772   2A  C3  FF         LHLD CURADR     ;GET ADDR OF LAST CHAR DONE
2541    0775   EB  .   .          XCHG
2542    0776   04  .   .          INR B        ;DOES LSTCOL = -1?
2543    0777   C2  7C  07         JNZ RCA260   ;NO
2544    077A   0D  .   .          DCR C        ;DECREMENT COLUMN COUNT
2545    077B   .   .   .   RCA255 EQU $
2546    077B   1B  .   .          DCX  D          ;SET TO NEXT DISPLAY BYTE
```

```
 2548    077C    .    .    .    ;****************************
 2549    077C    .    .    .    ; ROW HAS BEEN FOUND        *
 2550    077C    .    .    .    ; SEARCH FOR DESIRED COLUMN *
 2551    077C    .    .    .    ;****************************
 2552    077C    .    .    .    RCA260 EQU $
 2553    077C    21   FE   FF          LXI  H,DISPST  ;SET FOR NO CHARACTER MATCH
 2554    077F    CD   CF   1E          CALL FNDCHR    ;DOES CHARACTER EXIST?
 2555    0782    3E   00   .           MVI  A,DELTRM  ;SET FUNCTION FLAG TO DELETE
 2556    0784    32   6D   FF          STA  TRMFCT      ;NON-DISPLAYING TERMINATOR
 2557    0787    CC   69   09          CZ   EOLMVO    ;NO - TRY TO MOVE EOL
 2558    078A    EB   .    .           XCHG           ;SET NEW CURRENT CHAR ADDRES
 2559    078B    22   C3   FF          SHLD CURADR
 2560    078E    EB   .    .           XCHG
 2561    078F    21   C0   FF          LXI  H,CURROW
 2562    0792    46   .    .           MOV  B,M       ;GET DESIRED ROW AND COLUMN
 2563    0793    3A   85   FF          LDA  TMPCOL
 2564    0796    0D   .    .           DCR C         ;CONVERT TO COLUMN FOUND
 2565    0797    FA   9B   07          JM   RCA270
 2566    079A    91   .    .           SUB C
 2567    079B    .    .    .   RCA270 EQU $
 2568    079B    68   .    .           MOV  L,B       ;UPDATE LAST ROW AND COLUMN
 2569    079C    67   .    .           MOV  H,A         ;DONE
 2570    079D    22   C7   FF          SHLD LSTROW
 2571    07A0    26   FF   .           MVI  H,BASEH   ;SET H TO DATA PAGE
 2572    07A2    0C   .    .           INR C         ;RESTORE ZERO FLAG
 2573    07A3    C9   .    .           RET           ;RETURN
```

```
07A4    .   .   .    ;*****************************
07A4    .   .   .    ; TIMER INTERRUPT PROCESSING *
07A4    .   .   .    ;*****************************
07A4    .   .   .    ;
07A4    .   .   .    ;  ENTRY:  "PSW" AND B,C PUSHED
07A4    .   .   .    ;          A = INTERRUPT CODE
07A4    .   .   .    ;
07A4    .   .   .    TMINTR EQU $
07A4    CD  65  91          CALL INTVEC     ;TRY ALTERNATE INTERRUPT
07A7    3A  F5  FF          LDA  PRCCTL     ;GET PROCESSOR STATE
07AA    D5  .   .           PUSH D          ;SAVE REMAINING REGISTERS
07AB    E5  .   .           PUSH H
07AC    E6  FD  .           ANI  377Q-TMIEN
07AE    D3  70  .           OUT  PROCSR     ;ACKNOWLEDGE TIMER INTERRUPT
07B0    F6  02  .           ORI  TMIEN
07B2    D3  70  .           OUT  PROCSR     ;RE-ENABLE THE TIMER
07B4    21  D0  FF          LXI  H,RSTTMR   ;DECREMENT SOFT RESET DELAY
07B7    7E  .   .           MOV  A,M          ;TIMER
07B8    3D  .   .           DCR  A          ;COUNTING DOWN?
07B9    FA  C2  07          JM   TMI010     ;NO - DON'T UPDATE TIMER
07BC    77  .   .           MOV  M,A        ;YES - STORE NEW VALUE
07BD    3E  06  .           MVI  A,ENDTST     ;(SET FOR RESET LED'S)
07BF    CC  08  48          CZ   ZKBCTL     ;RESET LED'S IF TIME DONE
07C2    .   .   .    TMI010 EQU  $
07C2    2E  50  .           MVI  L,TPSTAL-BASE  ;DECREMENT TAPE STALLED
07C4    7E  .   .           MOV  A,M            ;COUNTER
07C5    3D  .   .           DCR  A          ;STALL LIMIT REACHED?
07C6    FA  CA  07          JM   TMI020     ;YES - DON'T UPDATE COUNTER
07C9    77  .   .           MOV  M,A        ;NO - STORE NEW VALUE
07CA    .   .   .    TMI020 EQU  $
07CA    2E  52  .           MVI  L,CTBLTM-BASE  ;DECREMENT BLINK TIMER
07CC    35  .   .           DCR  M          ;TIME OUT?
07CD    C2  DB  07          JNZ  TMI100     ;NO - EXIT
07D0    36  20  .           MVI  M,CTBDLY   ;YES - RESET TIMER
07D2    23  .   .           INX  H
07D3    7E  .   .           MOV  A,M        ;GET CTU BLINK MASK
07D4    2E  55  .           MVI  L,CMND-BASE
07D6    AE  .   .           XRA  M          ;TOGGLE BLINKING LIGHTS
07D7    77  .   .           MOV  M,A        ;UPDATE LIGHT STATE
07D8    32  00  8B          STA  IOCTCO     ;SET CTU LIGHTS
```

```
 2616    07DB    .    .    .    ;*****************************************
 2617    07DB    .    .    .    ; PERFORM KEYBOARD AND DATA COMM MONITOR *
 2618    07DB    .    .    .    ; ROUTINES                               *
 2619    07DB    .    .    .    ;*****************************************
 2620    07DB    .    .    .    TMI100 EQU  $
 2621    07DB    21   F6   FF          LXI  H,INTFLG  ;GET INTERRUPT FLAG
 2622    07DE    3E   04   .           MVI  A,TMRINT+1  ;TIMER INTERRUPT ALREADY
 2623    07E0    BE   .    .           CMP  M           ;IN PROGRESS?
 2624    07E1    CA   F7   07          JZ   TMI110    ;YES - DON'T DO MONITOR CALL
 2625    07E4    77   .    .           MOV  M,A       ;NO - SET IN-PROGRESS FLAG
 2626    07E5    3A   7F   FE          LDA  DEVFLG    ;GET DEVICE FLAGS
 2627    07E8    87   .    .           ADD  A         ;ALTERNATE I/O INSTALLED?
 2628    07E9    FC   0B   60          CM   ZMONAL    ;YES - MONITOR ALT DEVICE
 2629    07EC    CD   0B   48          CALL ZKBMON
 2630    07EF    F3   .    .           DI             ;****************************
 2631    07F0    CD   0E   50          CALL ZDCMON    ;* KEYBOARD MONITOR ROUTINE
 2632    07F3    .    .    .    ;                     * RE-ENABLES INTERRUPTS    *
 2633    07F3    .    .    .    ;                     ****************************
 2634    07F3    21   F6   FF          LXI  H,INTFLG  ;SET INTERUPT CODE TO
 2635    07F6    35   .    .           DCR  M           ;INDICATE TIMER INTERRUPT
 2636    07F7    .    .    .    TMI110 EQU  $
 2637    07F7    E1   .    .           POP  H         ;RESTORE CONTENTS OF
 2638    07F8    D1   .    .           POP  D           ;ALL REGISTERS AND
 2639    07F9    C1   .    .           POP  B           ;ALL CONDITION FLAGS
 2640    07FA    F1   .    .           POP  PSW
 2641    07FB    FB   .    .           EI             ;RE-ENABLE INTERRUPTS
 2642    07FC    C9   .    .           RET            ;RETURN TO NORMAL PROCESSING
```

```
=============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=============================================================================
 2644   07FD   .   .   .   ;**************************
 2645   07FD   .   .   .   ;  R O M   B R E A K   1  *
 2646   07FD   .   .   .   ;**************************
 2647   07FD   .   .   .          ORG   BEGIN+4000Q
 2648   0800   .   .   .   ZBRK1  EQU   $
 2649   0800   51  .   .          DB    VERSN1     ;ROM PRESENT FLAGS
 2650   0801   08  .   .          DB    ZBRK1/256
```

```
2652   0802   .    .    .   ;*****************************************
2653   0802   .    .    .   ; BINOCT - CONVERT BINARY TO OCTAL ASCII *
2654   0802   .    .    .   ;*****************************************
2655   0802   .    .    .   ;
2656   0802   .    .    .   ;  ENTRY:  A = DIGIT TO BE CONVERTED
2657   0802   .    .    .   ;          H,L = ADDRESS OF OUTPUT BUFFER'S
2658   0802   .    .    .   ;            HIGH ORDER BYTE
2659   0802   .    .    .   ;
2660   0802   .    .    .   ;  EXIT :  H,L = H,L(ENTRY)+4
2661   0802   .    .    .   ;          A-C DESTROYED
2662   0802   .    .    .   ;
2663   0802   .    .    .   ;  FIRST BYTE IS SET TO BLANK.  THE NEXT THREE
2664   0802   .    .    .   ;  BYTES CONTAIN THE ASCII OCTAL EQUIVALENT OF
2665   0802   .    .    .   ;  THE INPUT VALUE.  THE FIFTH BYTE IS SET TO
2666   0802   .    .    .   ;  ZERO (NULL).
2667   0802   .    .    .   ;
2668   0802   .    .    .   BINOCT EQU  $
2669   0802   36   20   .          MVI  M,ABLNK     ;SET FIRST BYTE TO BLANK
2670   0804   23   .    .          INX  H
2671   0805   06   03   .          MVI  B,3         ;SET B TO NUMBER OF DIGITS
2672   0807   07   .    .          RLC              ;ROTATE DOWN TWO HIGH ORDER
2673   0808   07   .    .          RLC                ;BITS
2674   0809   4F   .    .          MOV  C,A         ;SAVE VALUE IN C-REGISTER
2675   080A   E6   03   .          ANI  3Q          ;MASK OUT TWO HIGH ORDER BIT
2676   080C   .    .    .   BN0010 EQU  $
2677   080C   E6   07   .          ANI  7Q          ;MASK OUT NEXT THREE BITS
2678   080E   F6   30   .          ORI  ZERO        ;ADD IN ASCII ADJUSTMENT
2679   0810   77   .    .          MOV  M,A         ;STORE ASCII CHARACTER
2680   0811   23   .    .          INX  H           ;INCREMENT TO NEXT BYTE
2681   0812   79   .    .          MOV  A,C         ;RECALL INPUT
2682   0813   07   .    .          RLC              ;ROTATE  TO NEXT THREE BITS
2683   0814   07   .    .          RLC
2684   0815   07   .    .          RLC
2685   0816   4F   .    .          MOV  C,A         ;SAVE VALUE
2686   0817   05   .    .          DCR  B           ;ALL BITS DONE?
2687   0818   C2   0C   08         JNZ  BN0010      ;NO - SET NEXT BYTE
2688   081B   70   .    .          MOV  M,B         ;YES - STORE NULL IN BUFFER
2689   081C   C9   .    .          RET              ;RETURN
```

```
===========================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
===========================================================================
2691    081D   .    .    .   ;**********************************
2692    081D   .    .    .   ; BN2DE0 - CONVERT SINGLE BYTE TO *
2693    081D   .    .    .   ;    ASCII DECIMAL                *
2694    081D   .    .    .   ;**********************************
2695    081D   .    .    .   ;
2696    081D   .    .    .   ;   ENTRY:  A = BYTE TO BE CONVERTED
2697    081D   .    .    .   ;           H,L = ADDRESS OF OUTPUT BUFFER'S
2698    081D   .    .    .   ;             HIGH ORDER ADDRESS
2699    081D   .    .    .   ;
2700    081D   .    .    .   ;   EXIT :  NZ
2701    081D   .    .    .   ;           H,L = H,L(ENTRY)+3
2702    081D   .    .    .   ;           A-E DESTROYED
2703    081D   .    .    .   ;
2704    081D   .    .    .   BN2DE0 EQU  $
2705    081D   22   96   FF         SHLD LNKSAV      ;SAVE BUFFER ADDRESS
2706    0820   21   6A   08         LXI  H,B2D200    ;SET OUTPUT ROUTINE TO BUFFE
2707    0823   .    .    .   BN2DE1 EQU  $             ;STORE ROUTINE
2708    0823   22   CE   FF         SHLD CNTFAD      ;SET OUTPUT ROUTINE ADDRESS
2709    0826   .    .    .   BN2DE2 EQU  $            ;ENTRY FOR "ASCOUT"
2710    0826   5F   .    .          MOV  E,A         ;CHANGE INPUT INTO DOUBLE
2711    0827   16   00   .          MVI  D,0           ;BYTE VALUE
2712    0829   0E   01   .          MVI  C,1         ;SET ZERO SUPPRESS FLAG
2713    082B   C3   45   08         JMP  B2D050      ;GO TO CONVERT ROUTINE
```

```
 2715   082E    .    .    .   ;*************************************************
 2716   082E    .    .    .   ; BN2DEC - CONVERT DOUBLE WORD BINARY TO DECIMAL *
 2717   082E    .    .    .   ;*************************************************
 2718   082E    .    .    .   ;
 2719   082E    .    .    .   ;  ENTRY:  D,E = BINARY VALUE
 2720   082E    .    .    .   ;          H,L = ADDRESS OF HIGH ORDER BYTE IN
 2721   082E    .    .    .   ;            BUFFER
 2722   082E    .    .    .   ;
 2723   082E    .    .    .   ;  EXIT :  H,L = H,L(ENTRY)+5
 2724   082E    .    .    .   ;          A-E DESTROYED
 2725   082E    .    .    .   ;          LNKSAV DESTROYED
 2726   082E    .    .    .   ;
 2727   082E    .    .    .   ;  THE FIRST FIVE BYTES OF THE BUFFER CONTAIN THE
 2728   082E    .    .    .   ;  ASCII DECIMAL VALUE.  THE SIXTH BYTE IS SET TO
 2729   082E    .    .    .   ;  ZERO (NULL).  LEADING ZEROES ARE BLANKED
 2730   082E    .    .    .   ;
 2731   082E    .    .    .   BN2DEC EQU  $
 2732   082E   22   96   FF         SHLD LNKSAV     ;SAVE BUFFER ADDRESS
 2733   0831   21   6A   08         LXI  H,B2D200   ;SET OUTPUT ROUTINE TO BUFFE
 2734   0834   22   CE   FF         SHLD CNTFAD       ;STORE ROUTINE
 2735   0837   0E   01   .          MVI  C,1        ;SET ZERO SUPPRESS FLAG
 2736   0839   21   F0   D8         LXI  H,-10000
 2737   083C   CD   58   08         CALL B2D100     ;EXTRACT 10,000'S VALUE
 2738   083F   21   18   FC         LXI  H,-1000
 2739   0842   CD   58   08         CALL B2D100     ;EXTRACT 1,000'S VALUE
 2740   0845    .    .    .   B2D050 EQU  $
 2741   0845   21   9C   FF         LXI  H,-100
 2742   0848   CD   58   08         CALL B2D100     ;EXTRACT 100'S VALUE
 2743   084B   21   F6   FF         LXI  H,-10
 2744   084E   CD   58   08         CALL B2D100     ;EXTRACT 10'S VALUE
 2745   0851   7B    .    .         MOV  A,E        ;CONVERT UNITS DIGIT TO
 2746   0852   F6   30    .         ORI  ZERO         ;ASCII AND STORE IN
 2747   0854   0D    .    .         DCR  C          ;SET C TO FORCE ZERO STORE
 2748   0855   C3   CD   FF         JMP  ECONTF     ;GO TO OUTPUT ROUTINE
```

```
===============================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS
===============================================================================
2750    0858    .    .    .    ;******************************
2751    0858    .    .    .    ; B2D100 - EXTRACT RADIX VALUE *
2752    0858    .    .    .    ;******************************
2753    0858    .    .    .    ;
2754    0858    .    .    .    ;  ENTRY:  C = 0, SUPPRESS ZERO
2755    0858    .    .    .    ;            < 0, DON'T SUPPRESS ZEROES
2756    0858    .    .    .    ;          D,E = VALUE TO BE CONVERTED
2757    0858    .    .    .    ;          H,L = -RADIX
2758    0858    .    .    .    ;          LNKSAV = CURRENT BUFFER ADDRESS
2759    0858    .    .    .    ;
2760    0858    .    .    .    ;  EXIT :  C < 0, CHARACTER STORED
2761    0858    .    .    .    ;            = 0, ZERO SUPPRESSED
2762    0858    .    .    .    ;          (LNKSAV) = (LNKSAV)+1
2763    0858    .    .    .    ;          A-C, H,L DESTROYED
2764    0858    .    .    .    ;
2765    0858    .    .    .    B2D100 EQU  $
2766    0858    06   2F   .           MVI  B,ZERO-1     ;SET INITIAL ASCII VALUE
2767    085A    EB   .    .           XCHG              ;EXCHANGE RADIX AND INPUT
2768    085B    .    .    .    B2D110 EQU  $
2769    085B    04   .    .           INR  B            ;INCREMENT ASCII VALUE
2770    085C    19   .    .           DAD  D            ;SUBTRACT RADIX
2771    085D    DA   5B   08          JC   B2D110       ;CONTINUE IF INPUT>RADIX
2772    0860    7D   .    .           MOV  A,L          ;ADD BACK RADIX TO EXTRACT
2773    0861    93   .    .           SUB  E              ;REMAINDER
2774    0862    5F   .    .           MOV  E,A          ;SAVE REMAINDER IN D,E
2775    0863    7C   .    .           MOV  A,H
2776    0864    9A   .    .           SBB  D
2777    0865    57   .    .           MOV  D,A
2778    0866    78   .    .           MOV  A,B          ;GET CONVERTED VALUE
2779    0867    C3   CD   FF          JMP  ECONTF       ;GO TO OUTPUT ROUTINE
2780    086A    .    .    .    ;**********************************************
2781    086A    .    .    .    ; B2D200 - STORE DECIMAL VALUE FOR INTERNAL USE *
2782    086A    .    .    .    ;**********************************************
2783    086A    .    .    .    ;
2784    086A    .    .    .    ;  ENTRY:  A = CONVERTED VALUE
2785    086A    .    .    .    ;
2786    086A    .    .    .    B2D200 EQU  $
2787    086A    FE   30   .           CPI  ZERO         ;CONVERTED VALUE = ZERO?
2788    086C    C2   75   08          JNZ  B2D210       ;NO - STORE THE DIGIT
2789    086F    0D   .    .           DCR  C            ;NON-ZERO CHAR ALREADY DONE?
2790    0870    FA   76   08          JM   B2D220       ;YES - STORE THE DIGIT
2791    0873    0C   .    .           INR  C            ;NO - RESTORE ZERO FLAG
2792    0874    C9   .    .           RET                 ;AND EXIT
2793    0875    .    .    .    B2D210 EQU  $
2794    0875    0D   .    .           DCR  C            ;CLEAR ZERO SUPPRESS FLAG
2795    0876    .    .    .    B2D220 EQU  $
2796    0876    2A   96   FF          LHLD LNKSAV       ;GET BUFFER POINTER
2797    0879    77   .    .           MOV  M,A          ;STORE CONVERTED VALUE
2798    087A    23   .    .           INX  H            ;INCREMENT BUFFER POINTER
2799    087B    36   00   .           MVI  M,0          ;SET NEXT BYTE TO NULL
2800    087D    22   96   FF          SHLD LNKSAV       ;STORE NEW POINTER VALUE
2801    0880    C9   .    .           RET               ;RETURN
```

```
 2803    0881    .    .    .   ;**********************************
 2804    0881    .    .    .   ; CALCULATE CHECKSUM               *
 2805    0881    .    .    .   ;                                  *
 2806    0881    .    .    .   ; ENTRY:                           *
 2807    0881    .    .    .   ;        (H,L) = ADDRESS OF AREA   *
 2808    0881    .    .    .   ;        TO BE CHECKSUMED          *
 2809    0881    .    .    .   ;                                  *
 2810    0881    .    .    .   ;        D = NO. BYTES IN AREA/256 *
 2811    0881    .    .    .   ; WE ASSUME THE AREA BEGINS ON A  *
 2812    0881    .    .    .   ; 256 BYTE BOUNDARY, I.E., L=0.   *
 2813    0881    .    .    .   ;        CALL CHKSUM               *
 2814    0881    .    .    .   ; EXIT:                            *
 2815    0881    .    .    .   ;        A = CHECKSUM              *
 2816    0881    .    .    .   ;        ALL OTHER REGS. UNCHANGED *
 2817    0881    .    .    .   ;        FLAGS DESTROYED           *
 2818    0881    .    .    .   ;**********************************
 2819    0881    .    .    .   CHKSUM EQU  $
 2820    0881    D5   .    .          PUSH D           ;SAVE REGISTER D-H
 2821    0882    E5   .    .          PUSH H
 2822    0883    AF   .    .          XRA  A           ;ZERO SUM
 2823    0884    .    .    .   CSU100 EQU  $
 2824    0884    86   .    .          ADD  M           ;ADD BYTE
 2825    0885    CE   00   .          ACI  0           ;ADD CARRY
 2826    0887    2C   .    .          INR  L           ;BUMP ADDRESS POINTER
 2827    0888    C2   84   08         JNZ  CSU100      ;ADD NEXT BYTE
 2828    088B    .    .    .   ;
 2829    088B    24   .    .          INR  H           ;FINISHED A 256 BYTE BLOCK
 2830    088C    15   .    .          DCR  D
 2831    088D    C2   84   08         JNZ  CSU100      ;DO NEXT 256 BYTES
 2832    0890    .    .    .   ;
 2833    0890    03   .    .          INX  B           ;INCREMENT TO NEXT STORE ADD
 2834    0891    57   .    .          MOV  D,A         ;SAVE CHECKSUM IN D-REGISTER
 2835    0892    E1   .    .          POP  H           ;RECALL STARTING ADDRESS
 2836    0893    7C   .    .          MOV  A,H
 2837    0894    FE   F0   .          CPI  170000Q/256  ;LAST RAM BLOCK?
 2838    0896    C2   9A   08         JNZ  CSU110      ;NO - EXIT
 2839    0899    4D   .    .          MOV  C,L         ;YES - SET B,C TO FIRST
 2840    089A    .    .    .   ;                         CHECKSUM STORE ADDRESS
 2841    089A    .    .    .   CSU110 EQU  $
 2842    089A    7A   .    .          MOV  A,D         ;PUT CHECKSUM BACK INTO A-RE
 2843    089B    D1   .    .          POP  D           ;RESTORE D,E
 2844    089C    C9   .    .          RET              ;RETURN
 2846    089D    .    .    .    ;*******************************************
 2847    089D    .    .    .    ; CLEAR - RESET TERMINAL BY ESCAPE SEQUENCE *
 2848    089D    .    .    .    ;*******************************************
 2849    089D    .    .    .    CLEAR   EQU  $
 2850    089D    CD   6E   15           CALL IOBSYC     ;WAIT UNTIL TAPES NOT BUSY
 2851    08A0    3E   04   .            MVI  A,FRCRST   ;SET FLAG TO FORCE FULL
 2852    08A2    CD   00   14           CALL STCMFL       ;TERMINAL RESET
 2853    08A5    3E   80   .            MVI  A,CRTOFF   ;TURN OFF THE DISPLAY
 2854    08A7    32   20   87           STA  IOCRRW
 2855    08AA    C7   .    .            RST  ;RESET      GO DO TERMINAL RESET
```

```
=====================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                              PAGE  82
=====================================================================================
 2857    08AB   .   .   .   ;*******************************************
 2858    08AB   .   .   .   ; DISPL1 - ADD ENOUGH BLOCKS TO REACH DESIRED *
 2859    08AB   .   .   .   ;   COLUMN                                    *
 2860    08AB   .   .   .   ;*******************************************
 2861    08AB   .   .   .   ;
 2862    08AB   .   .   .   ;  ENTRY:  C = NUMBER OF CHARACTERS NEEDED - 1
 2863    08AB   .   .   .   ;          D,E = LOCATION OF EOL IN LINE
 2864    08AB   .   .   .   ;
 2865    08AB   .   .   .   ;  EXIT :  A = 0, NOT ENOUGH BLOCKS (MEMORY LOCK)
 2866    08AB   .   .   .   ;            B-L DESTROYED
 2867    08AB   .   .   .   ;          A # 0, MEMORY ALLOCATED
 2868    08AB   .   .   .   ;            D,E = FIRST CHAR ADDR IN NEW BLOCKS
 2869    08AB   .   .   .   ;            B,C,H,L DESTROYED
 2870    08AB   .   .   .   ;
 2871    08AB   .   .   .   ;  IF ONLY ONE CHARACTER IS TO BE ADDED, THE
 2872    08AB   .   .   .   ;  CHARACTER IS ADDED TO THE LINE.  OTHERWISE, ADD
 2873    08AB   .   .   .   ;  REQUIRED BLOCKS ARE ADDED TO THE LINE AND THE
 2874    08AB   .   .   .   ;  LINE IS FILLED WITH BLANKS UP TO THE DESIRED
 2875    08AB   .   .   .   ;  CHARACTER ONLY.
 2876    08AB   .   .   .   ;
 2877    08AB   .   .   .   DISPL1 EQU $
 2878    08AB   0C  .   .          INR C          ;MOVE EOL IF NECESSARY
 2879    08AC   CD  71  09         CALL EOLMOV
 2880    08AF   0D  .   .          DCR C
 2881    08B0   FA  4A  09         JM    DIS220    ;CHARACTER POSITION FOUND
 2882    08B3   21  96  FF         LXI   H,NCHAR   ;SAVE NUMBER OF CHARACTERS
 2883    08B6   71  .   .          MOV   M,C            ;TO BE ADDED - 1
 2884    08B7   .   .   .   DISPL2 EQU   $
 2885    08B7   EB  .   .          XCHG
 2886    08B8   22  94  FF         SHLD EOLADR   ;SAVE EOL ADDRESS
 2887    08BB   0D  .   .          DCR   C         ;SINGLE CHARACTER ADDED?
 2888    08BC   FA  5A  09         JM    DIS400    ;YES - DO FAST EXTEND
 2889    08BF   3E  20  .          MVI   A,ABLNK   ;NO - GET A DISPLAY BLOCK
 2890    08C1   CD  4D  05         CALL GTBLK        ;FILLED WITH BLANKS
 2891    08C4   C8  .   .          RZ              ;RETURN IF MEMORY LOCKED
 2892    08C5   EB  .   .          XCHG            ;PUT BLOCK ADDRESS IN D,E
 2893    08C6   F6  0F  .          ORI   BLKSM     ;COMPUTE HIGH ADDR OF BLOCK
 2894    08C8   4F  .   .          MOV   C,A       ;SAVE ADDRESS OF FIRST NEW
 2895    08C9   C5  .   .          PUSH B            ;BLOCK ADDED
 2896    08CA   3A  96  FF         LDA   NCHAR     ;GET # OF CHARS TO BE ADDED
 2897    08CD   06  00  .          MVI B,0        ;INITIALIZE COUNT
 2898    08CF   .   .   .   DIS120 EQU $
 2899    08CF   04  .   .          INR B          ;INCREMENT COUNT
 2900    08D0   D6  0E  .          SUI BLKSZ-2    ;SUB. NO. OF CHARS IN BLOCK
 2901    08D2   F2  CF  08         JP   DIS120    ;JUMP IF MORE BLOCKS NEEDED
 2902    08D5   32  84  FF         STA   COUNT      ;SAVE LAST CHAR BLOCK POS
 2903    08D8   05  .   .          DCR B          ;SINGLE BLOCK?
 2904    08D9   CA  FA  08         JZ   DIS160    ;YES
```

```
=========================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                    PAGE  83
=========================================================================
2906   08DC    .   .   .    ;***************************
2907   08DC    .   .   .    ; MULTIPLE BLOCKS REQUIRED *
2908   08DC    .   .   .    ;***************************
2909   08DC    21  99  FF          LXI  H,NBLKS   ;SAVE BLOCK COUNT
2910   08DF    70  .   .           MOV M,B
2911   08E0    .   .   .    ;************************
2912   08E0    .   .   .    ; GET SUBSEQUENT BLOCKS *
2913   08E0    .   .   .    ;************************
2914   08E0    D5  .   .           PUSH D         ;SAVE ADDRESS OF LAST BLOCK
2915   08E1    .   .   .    DIS140 EQU  $
2916   08E1    3E  20  .           MVI  A,ABLNK   ;GET A DISPLAY BLOCK FILLED
2917   08E3    CD  4D  05          CALL GTBLK       ;WITH BLANKS
2918   08E6    EB  .   .           XCHG           ;PUT BLOCK ADDRESS IN D,E
2919   08E7    E1  .   .           POP  H         ;RECALL ADDRESS OF LAST BLOC
2920   08E8    CA  54  09          JZ   DIS240    ;EXIT IF MEMORY LOCKED
2921   08EB    D5  .   .           PUSH D         ;SAVE NEW LINE ADDRESS
2922   08EC    2B  .   .           DCX  H         ;LINK NEW BLOCK TO PREVIOUS
2923   08ED    F6  0F  .           ORI BLKSM
2924   08EF    70  .   .           MOV M,B      ;MSB'S
2925   08F0    2B  .   .           DCX  H
2926   08F1    77  .   .           MOV  M,A       ;STORE LSB
2927   08F2    21  99  FF          LXI  H,NBLKS
2928   08F5    35  .   .           DCR  M         ;ALL BLOCKS ALLOCATED?
2929   08F6    C2  E1  08          JNZ  DIS140    ;NO - GET ANOTHER BLOCK
2930   08F9    F1  .   .           POP  PSW       ;YES - POP THE STACK
```

```
=====================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                   PAGE  84
=====================================================================
2932    08FA   .    .    .   ;****************************
2933    08FA   .    .    .   ; ALL BLOCKS HAVE BEEN ADDED *
2934    08FA   .    .    .   ;****************************
2935    08FA   .    .    .   DIS160 EQU  $
2936    08FA   3A   84   FF         LDA  COUNT      ;COMPUTE NUMBER OF BYTES
2937    08FD   2F   .    .          CMA               ;TO FILL
2938    08FE   3C   .    .          INR  A
2939    08FF   4F   .    .          MOV C,A       ;SAVE IN C
2940    0900   83   .    .          ADD E         ;GET FIRST FILL ADDR
2941    0901   3D   .    .          DCR  A          ;SET FIRST LSB FILL ADDRESS
2942    0902   6F   .    .          MOV  L,A        ;PUT LSB INTO L
2943    0903   62   .    .          MOV  H,D        ;GET MSB FROM D
2944    0904   06   CC   .          MVI  B,EOL      ;SET "EOL" CHARACTER
2945    0906   3A   C1   FF         LDA  CURCOL     ;GET CURRENT COLUMN
2946    0909   FE   4F   .          CPI  MAXCOL     ;CHAR ADDED TO LAST COLUMN?
2947    090B   C2   15   09         JNZ  DIS170     ;NO - SET "EOL" CHARACTER
2948    090E   3A   89   FF         LDA  DCHAR      ;YES - GET CHARACTER STORED
2949    0911   B7   .    .          ORA  A          ;IS IT ASCII?
2950    0912   F2   16   09         JP   DIS175     ;YES - DON'T ADD "EOL"
2951    0915   .    .    .   ;                       NO - SET "EOL" CHARACTER
2952    0915   .    .    .   ;**********************************************
2953    0915   .    .    .   ; FILL UNUSED PART OF BLOCK WITH "FILL" CODES *
2954    0915   .    .    .   ;**********************************************
2955    0915   .    .    .   DIS170 EQU  $
2956    0915   70   .    .          MOV  M,B        ;STORE FILL/EOL CHARACTER
2957    0916   .    .    .   DIS175 EQU  $
2958    0916   2B   .    .          DCX  H          ;GO TO NEXT BYTE
2959    0917   0D   .    .          DCR  C          ;ALL UNUSED BYTES FILLED?
2960    0918   06   C3   .          MVI  B,FILL       ;(SET "FILL" CODE)
2961    091A   C2   15   09         JNZ  DIS170     ;NO - SET NEXT BYTE
2962    091D   .    .    .   ;*************************
2963    091D   .    .    .   ; WRITE LINK TO NEXT LINE *
2964    091D   .    .    .   ;*************************
2965    091D   .    .    .   DIS180 EQU $
2966    091D   2A   C9   FF         LHLD LSTLIN     ;GET ADDR CURRENT LINE
2967    0920   EB   .    .          XCHG
2968    0921   2B   .    .          DCX  H          ;STORE AS NEXT BLOCK POINTER
2969    0922   72   .    .          MOV  M,D
2970    0923   2B   .    .          DCX  H
2971    0924   13   .    .          INX  D          ;POINT TO NEXT LINE POINTER
2972    0925   73   .    .          MOV  M,E
```

```
2974   0926   .   .   .   ;****************************
2975   0926   .   .   .   ; LINK NEW BLOCK(S) TO OLD *
2976   0926   .   .   .   ;****************************
2977   0926   D1  .   .           POP  D         ;RECALL FIRST NEW BLOCK ADDR
2978   0927   3A  9B  FF          LDA  NCHAR     ;GET # OF CHARS ADDED - 1
2979   092A   B7  .   .           ORA  A         ;DOES NEW CHAR REPLACE EOL?
2980   092B   3A  89  FF          LDA  DCHAR      ;(DEFAULT TO ADD 1 CHAR)
2981   092E   CA  33  09          JZ   DIS210    ;YES - OVERWRITE EOL
2982   0931   3E  20  .           MVI  A,ABLNK   ;NO - STORE BLANK OVER EOL
2983   0933   .   .   .   DIS210 EQU $
2984   0933   47  .   .           MOV  B,A       ;SAVE CHARACTER TO BE STORED
2985   0934   2A  94  FF          LHLD EOLADR    ;RECALL EOL ADDRESS
2986   0937   3A  C0  FF          LDA  CURROW
2987   093A   F6  40  .           ORI  MAYEOL    ;SET FOR POSSIBLE EOL SKIP
2988   093C   F3  .   .           DI             ;DISABLE INTERRUPTS
2989   093D   32  20  87          STA  IOCRRW    ;TURN OFF DISPLAY DMA
2990   0940   70  .   .           MOV  M,B       ;OVERWRITE EOL
2991   0941   2B  .   .           DCX  H
2992   0942   72  .   .           MOV  M,D       ;CHANGE NEXT BLOCK LINK TO
2993   0943   2B  .   .           DCX  H           ;POINT TO NEW BLOCKS
2994   0944   73  .   .           MOV  M,E
2995   0945   CD  9E  0F          CALL DISDN1    ;TURN DISPLAY BACK ON
2996   0948   B4  .   .           ORA  H         ;SET Z-FALSE
2997   0949   C9  .   .           RET            ;RETURN
2998   094A   .   .   .   ;*****************************
2999   094A   .   .   .   ; EOL MOVE SATISFIED REQUEST *
3000   094A   .   .   .   ; CHECK FOR SINGLE CHARACTER *
3001   094A   .   .   .   ;*****************************
3002   094A   .   .   .   DIS220 EQU $
3003   094A   3D  .   .           DCR  A         ;SINGLE CHARACTER?
3004   094B   32  9B  FF          STA  NCHAR     ;(SET NCHAR)
3005   094E   C0  .   .           RNZ            ;NO - RETURN
3006   094F   3A  89  FF          LDA  DCHAR     ;YES - GET THE CHARACTER
3007   0952   12  .   .           STAX D         ;STORE THE CHARACTER
3008   0953   C9  .   .           RET          ;RETURN
3009   0954   .   .   .   ;***************************
3010   0954   .   .   .   ; ALL BLOCKS NOT AVAILABLE *
3011   0954   .   .   .   ; INITIALIZE END OF LINE   *
3012   0954   .   .   .   ;***************************
3013   0954   .   .   .   DIS240 EQU $
3014   0954   36  CC  .           MVI  M,EOL     ;STORE AN EOL
3015   0956   EB  .   .           XCHG           ;PUT ADDRESS INTO D,E
3016   0957   C3  1D  09          JMP DIS180
```

```
==============================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                          PAGE  86
==============================================================================
 3018    095A   .   .   .    ;****************************
 3019    095A   .   .   .    ; SINGLE CHARACTER ADDITION *
 3020    095A   .   .   .    ;****************************
 3021    095A   .   .   .    DIS400 EQU  $
 3022    095A   CD  4B  05          CALL GTBLKF     ;GET A DISPLAY BLOCK
 3023    095D   C8  .   .           RZ              ;RETURN IF MEMORY LOCKED
 3024    095E   54  .   .           MOV  D,H        ;SAVE BLOCK ADDRESS IN D,E
 3025    095F   5D  .   .           MOV  E,L
 3026    0960   F6  0F  .           ORI  BLKSM      ;PUT AN EOL AT THE FIRST
 3027    0962   6F  .   .           MOV  L,A          ;DISPLAY CHARACTER
 3028    0963   36  CC  .           MVI  M,EOL        ;LOCATION IN THE BLOCK
 3029    0965   E5  .   .           PUSH H          ;SAVE ADDRESS OF BLOCK
 3030    0966   C3  1D  09          JMP  DIS180     ;LINK BLOCK TO DISPLAY
```

```
===========================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                          PAGE  87
===========================================================================
3032    0969    .    .    .  ;
3033    0969    .    .    .  ; * * * * * * * * * * * * * * * * * * * * * * *
3034    0969    .    .    .  ;
3035    0969    .    .    .  ;  EOLMOV - MOVE EOL IN A BLOCK
3036    0969    .    .    .  ;
3037    0969    .    .    .  ;       ENTRY:  C = NUMBER OF BYTES NEEDED
3038    0969    .    .    .  ;               D,E = ADDRESS OF EXISTING EOL
3039    0969    .    .    .  ;
3040    0969    .    .    .  ;       EXIT :  A = NUMBER OF CHARACTERS ADDED
3041    0969    .    .    .  ;               C = 0, CHARACTER FOUND
3042    0969    .    .    .  ;                 D,E = ADDRESS OF CHARACTER
3043    0969    .    .    .  ;               C = NUMBER OF CHARACTERS NEEDED
3044    0969    .    .    .  ;                 D,E = ADDRESS OF LAST BYTE IN BLK
3045    0969    .    .    .  ;               H = BASEH
3046    0969    .    .    .  ;               B,L DESTROYED
3047    0969    .    .    .  ;
3048    0969    .    .    .  ;  EOLMVO - MOVE ONLY IF UNPROTECTED
3049    0969    .    .    .  ;
3050    0969    .    .    .  EOLMVO EQU   $
3051    0969    3A   91   FF        LDA   BLKFIL       ;GET BLOCK FILL INHIBIT FLAG
3052    096C    3C   .    .         INR   A            ;BLOCK FILL INHIBITED OR
3053    096D    C4   65   10        CNZ   CKPROT         ;CURSOR IN PROTECTED FIELD
3054    0970    C8   .    .         RZ                 ;YES - RETURN
3055    0971    .    .    .  ;
3056    0971    .    .    .  EOLMOV EQU   $
3057    0971    7B   .    .         MOV   A,E          ;COMPUTE NUMBER OF BYTES
3058    0972    E6   0F   .         ANI   BLKSM          ;AVAILABLE IN BLOCK
3059    0974    D6   02   .         SUI   2              ;(DELETE BYTES FOR LINK)
3060    0976    C8   .    .         RZ                 ;RETURN IF NONE AVAILABLE
3061    0977    EB   .    .         XCHG               ;PUT CURRENT ADDRESS IN H,L
3062    0978    B9   .    .         CMP   C            ;ENOUGH CHARACTERS?
3063    0979    47   .    .         MOV   B,A            ;(SET B TO FILL BLOCK)
3064    097A    11   40   CC        LXI   D,EOL*256+MAYEOL  ;(SET FOR PARTIAL
3065    097D    .    .    .  ;                            LINE EXTENSION)
3066    097D    FA   92   09        JM    ELM100       ;NO - BLANK REST OF BLOCK
3067    0980    41   .    .         MOV   B,C          ;YES - BLANK WHAT'S NEEDED
3068    0981    3A   C1   FF        LDA   CURCOL       ;GET CURRENT COLUMN POSITION
3069    0984    FE   4F   .         CPI   MAXCOL       ;ADDING TO LAST COLUMN?
3070    0986    C2   92   09        JNZ   ELM100       ;NO - NEED EOL AT LINE END
3071    0989    3A   89   FF        LDA   DCHAR        ;YES - GET NEW CHARACTER
3072    098C    B7   .    .         ORA   A            ;IS IT  ASCII?
3073    098D    FA   92   09        JM    ELM100       ;NO - NEED EOL AT LINE END
3074    0990    16   C3   .         MVI   D,FILL       ;YES - DON'T NEED EOL
```

```
==========================================================================
 ITEM     LOC    OBJECT CODE  SOURCE STATEMENTS                    PAGE  88
==========================================================================
 3076    0992    .    .    .    ;
 3077    0992    .    .    .    ;  FILL THE BLOCK
 3078    0992    .    .    .    ;
 3079    0992    .    .    .    ELM100 EQU  $
 3080    0992    79   .    .           MOV  A,C         ;COMPUTE NUMBER OF ADDITIONA
 3081    0993    90   .    .           SUB  B             ;BYTES NEEDED
 3082    0994    4F   .    .           MOV  C,A         ;SAVE IT IN C FOR RETURN
 3083    0995    3A   C0   FF          LDA  CURROW      ;SET CONTROL TO TURN OFF DMA
 3084    0998    B3   .    .           ORA  E
 3085    0999    58   .    .           MOV  E,B         ;SAVE NUMBER OF BYTES ADDED
 3086    099A    F3   .    .           DI               ;DISABLE INTERRUPTS
 3087    099B    32   20   87          STA  IOCRRW   ;TURN OFF DMA
 3088    099E    .    .    .    ;
 3089    099E    .    .    .    ELM110 EQU  $
 3090    099E    36   20   .           MVI  M,ABLNK     ;FILL BLOCK WITH BLANKS
 3091    09A0    2B   .    .           DCX  H           ;MOVE TO NEXT BYTE
 3092    09A1    05   .    .           DCR  B           ;FILL COMPLETED?
 3093    09A2    C2   9E   09          JNZ  ELM110      ;NO - DO NEXT BYTE
 3094    09A5    72   .    .           MOV  M,D         ;YES - ADD EOL OR EOL FILL
 3095    09A6    CD   9E   0F          CALL DISLN1      ;TURN DISPLAY BACK ON
 3096    09A9    AF   .    .           XRA  A           ;CLEAR A-REGISTER
 3097    09AA    B1   .    .           ORA  C           ;ALL CHARACTERS DONE?
 3098    09AB    C2   AF   09          JNZ  ELM130      ;NO - RETURN ADDRESS OF EOL
 3099    09AE    23   .    .           INX  H           ;YES - RETURN ADDR OF LAST C
 3100    09AF    .    .    .    ELM130 EQU  $
 3101    09AF    7B   .    .           MOV  A,E         ;PUT # OF CHARS DONE IN A-RE
 3102    09B0    EB   .    .           XCHG             ;PUT CHARACTER ADDRESS IN D,
 3103    09B1    21   90   FF          LXI  H,EOLMV       ;(SET H TO DATA PAGE)
 3104    09B4    36   01   .           MVI  M,1         ;SET EOLMV FLAG
 3105    09B6    C9   .    .           RET              ;RETURN
```

```
===========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                       PAGE  89
===========================================================================
 3107    09B7    .    .    .    ;*******************
 3108    09B7    .    .    .    ; LD - LINE DELETE *
 3109    09B7    .    .    .    ;*******************
 3110    09B7    .    .    .    LINDEL EQU $
 3111    09B7    CD   76   19          CALL CHKFMS  ;FORMAT MODE?
 3112    09BA    CC   B4   06          CZ   RCADR1  ;FIND LINE IF NOT
 3113    09BD    C0   .    .           RNZ          ;LINE NOT FOUND
 3114    09BE    2A   C9   FF          LHLD LSTLIN    ;GET ADDR OF LAST LINE DONE
 3115    09C1    7E   .    .           MOV  A,M       ;GET PREVIOUS LINE'S LSB
 3116    09C2    B7   .    .           ORA  A         ;ANY PREVIOUS LINES?
 3117    09C3    CA   D4   09          JZ   LID050    ;NO - DO CLEAR LINE ONLY
 3118    09C6    CD   DA   09          CALL LINDL0    ;YES - DELETE CURRENT LINE
 3119    09C9    .    .    .    ;***************************************
 3120    09C9    .    .    .    ; UPDATE LSTLIN AND CURADR TO ADDRESS *
 3121    09C9    .    .    .    ; OF NEXT LINE                        *
 3122    09C9    .    .    .    ;***************************************
 3123    09C9    60   .    .           MOV  H,B       ;PUT NEW LINE ADDRESS INTO
 3124    09CA    69   .    .           MOV  L,C         ;H,L
 3125    09CB    CD   E8   18          CALL BACKT5    ;UPDATE CURRENT LINE AND ADD
 3126    09CE    CD   27   0A          CALL LININ0    ;GO UPDATE TOP LINE IF NEEDE
 3127    09D1    C3   91   06          JMP  PUTLIN    ;ADD LINE TO FREE LIST
 3128    09D4    .    .    .    LID050 EQU  $
 3129    09D4    CD   3C   1C          CALL CLEARL    ;CLEAR THE LINE
 3130    09D7    C3   C5   21          JMP  CURPRT    ;SET CURSOR AT LEFT MARGIN
 3132    09DA   .    .    .    ;***************************************
 3133    09DA   .    .    .    ; LINDLO - RMOVE LINE FROM LINKED LIST *
 3134    09DA   .    .    .    ;***************************************
 3135    09DA   .    .    .    ;
 3136    09DA   .    .    .    ;  ENTRY:  H,L = ADDRESS OF NEXT LINE FIELD
 3137    09DA   .    .    .    ;             (LSB) OF LINE TO BE DELETED
 3138    09DA   .    .    .    ;
 3139    09DA   .    .    .    ;  EXIT :  B,C = ADDRESS OF LSB PORTION OF
 3140    09DA   .    .    .    ;             NEXT LINE POINTER IN NEW LINE
 3141    09DA   .    .    .    ;          D,E = H,L(ENTRY)
 3142    09DA   .    .    .    ;          A,H,L DESTRUYED
 3143    09DA   .    .    .    ;
 3144    09DA   .    .    .    LINDLO EQU  $
 3145    09DA   5D   .    .           MOV  E,L        ;SAVE ADDRESS OF LINE TO BE
 3146    09DB   54   .    .           MOV  D,H          ;DELETED IN D,E
 3147    09DC   4E   .    .           MOV  C,M        ;GET ADDRESS OF NEXT LINE
 3148    09DD   23   .    .           INX  H
 3149    09DE   46   .    .           MOV  B,M
 3150    09DF   23   .    .           INX  H          ;GET ADDRESS OF PREVIOUS LIN
 3151    09E0   7E   .    .           MOV  A,M
 3152    09E1   23   .    .           INX  H
 3153    09E2   66   .    .           MOV  H,M
 3154    09E3   B7   .    .           ORA  A          ;DOES PREVIOUS LINE EXIST?
 3155    09E4   C2   F0   09          JNZ  LID200     ;YES - LINK 2 LINES TOGETHER
 3156    09E7   .    .    .    ;**************************************
 3157    09E7   .    .    .    ;  FIRST LINE DELETED - UPDATE FLINE  *
 3158    09E7   .    .    .    ;**************************************
 3159    09E7   60   .    .           MOV  H,B        ;MOVE NEW CURRENT LINE TO H,
 3160    09E8   69   .    .           MOV  L,C
 3161    09E9   23   .    .           INX  H          ;SET ADDR TO NEXT LINE FIELD
 3162    09EA   22   9F   FF          SHLD FLINE
 3163    09ED   C3   F6   09          JMP  LID300     ;SET NEW PREV LINE POINTER
```

```
==============================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                          PAGE  91
==============================================================================
3165   09F0    .    .    .   ;******************************************
3166   09F0    .    .    .   ; UPDATE NEXT LINE FIELD IN PREVIOUS LINE *
3167   09F0    .    .    .   ;******************************************
3168   09F0    .    .    .   LID200 EQU $
3169   09F0    6F   .    .          MOV  L,A        ;PUT LSB INTO L-REGISTER
3170   09F1    23   .    .          INX  H          ;SET TO MSB OF NEXT LINE FLD
3171   09F2    CD   95   0F         CALL DISLNK     ;SET NEW NEXT LINE LINK TO
3172   09F5    .    .    .   ;                          CURRENT ROW
3173   09F5    .    .    .   ;****************************************
3174   09F5    .    .    .   ; SET PREVIOUS LINE FIELD IN NEXT LINE *
3175   09F5    .    .    .   ;****************************************
3176   09F5    7D   .    .          MOV  A,L        ;SAVE PREV LINE ADDR'S LSB
3177   09F6    .    .    .   LID300 EQU  $
3178   09F6    03   .    .          INX  B          ;INCREMENT TO NEXT LINE PTR
3179   09F7    C5   .    .          PUSH B          ;SAVE ADDRESS
3180   09F8    03   .    .          INX  B          ;SET ADDRESS TO PREVIOUS
3181   09F9    03   .    .          INX  B            ;LINE FIELD
3182   09FA    02   .    .          STAX B          ;STORE LSB VALUE
3183   09FB    03   .    .          INX  B
3184   09FC    7C   .    .          MOV  A,H
3185   09FD    02   .    .          STAX B          ;STORE MSB VLAUE
3186   09FE    C1   .    .          POP  B          ;RESTORE CONTENTS OF B,C
3187   09FF    C9   .    .          RET             ;RETURN
```

```
===============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                              PAGE  92
===============================================================================
 3189   0A00   .    .    .   ;*******************
 3190   0A00   .    .    .   ; LI - LINE INSERT *
 3191   0A00   .    .    .   ;*******************
 3192   0A00   .    .    .   LININS EQU $
 3193   0A00   CD   76   19        CALL CHKFMS   ;FORMAT MODE?
 3194   0A03   CC   B4   06        CZ  RCADR1    ;FIND LINE IF NOT
 3195   0A06   C0   .    .         RNZ           ;RETURN IF LINE NOT FOUND
 3196   0A07   CD   4B   05        CALL GTBLKF   ;GET BLOCK FOR NEW LINE
 3197   0A0A   C8   .    .         RZ            ;RETURN IF NOT AVAILABLE
 3198   0A0B   .    .    .   ;*********************************
 3199   0A0B   .    .    .   ; STORE LINK AT END OF NEW LINE *
 3200   0A0B   .    .    .   ;*********************************
 3201   0A0B   C6   0B   .         ADI BLKSZ-5   ;GET ADDR OF NEXT LINE FIELD
 3202   0A0D   2D   .    .         DCR L
 3203   0A0E   74   .    .         MOV M,H       ;STORE LINK MSB'S
 3204   0A0F   2D   .    .         DCR L
 3205   0A10   77   .    .         MOV M,A       ;STORE LINK LSB'S
 3206   0A11   D6   02   .         SUI 2         ;STORE EOL IN NEW LINE
 3207   0A13   6F   .    .         MOV L,A
 3208   0A14   CD   68   0D        CALL STCHR1     ;SET FIRST DISPLAY CHARACTER
 3209   0A17   .    .    .   ;**********************************************
 3210   0A17   .    .    .   ; ADJUST LSTLIN AND CURADR PNTRS TO NEW LINE *
 3211   0A17   .    .    .   ;**********************************************
 3212   0A17   22   C3   FF        SHLD CURADR     ;SET CURADR TO 1ST CHAR
 3213   0A1A   23   .    .         INX  H          ;SET TO NEXT LINE POINTER
 3214   0A1B   7D   .    .         MOV  A,L        ;PUT LSB INTO A-REGISTER
 3215   0A1C   EB   .    .         XCHG
 3216   0A1D   2A   C9   FF        LHLD LSTLIN     ;GET CURRENT LINE ADDRESS
 3217   0A20   EB   .    .         XCHG
 3218   0A21   22   C9   FF        SHLD LSTLIN     ;SET NEW CURRENT LINE ADDRES
 3219   0A24   CD   3C   0A        CALL LININ1     ;ADD LINE TO DISPLAY LIST
 3220   0A27   .    .    .   ;****************************
 3221   0A27   .    .    .   ; UPDATE TOPLIN IF ROW ZERO *
 3222   0A27   .    .    .   ;****************************
 3223   0A27   .    .    .   LININO EQU  $
 3224   0A27   CD   A0   0A        CALL LSTLU2     ;SET INITIAL LINE STATE
 3225   0A2A   CD   C5   21        CALL CURPRT     ;SET CURSOR TO LEFT MARGIN
 3226   0A2D   AF   .    .         XRA  A          ;SET LAST COLUMN DONE TO
 3227   0A2E   32   C8   FF        STA  LSTCOL       ;ZERO
 3228   0A31   21   C0   FF        LXI  H,CURROW   ;GET CURRENT ROW NUMBER
 3229  .0A34   B6   .    .         ORA  M          ;DID TOP ROW CHANGE?
 3230   0A35   C0   .    .         RNZ             ;NO - RETURN
 3231   0A36   C3   86   0F        JMP  TOPUP1     ;YES - UPDATE TOP LINE VALUE
```

```
==========================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                    PAGE  93
==========================================================================
3233    0A39    .    .    .    ;******************************************
3234    0A39    .    .    .    ; LININ1 - ADD LINE TO LINK LIST           *
3235    0A39    .    .    .    ; ENTRY: D,E=NEXT PAGE FIELD ADDR IN LINE *
3236    0A39    .    .    .    ;            BEFORE WHICH NEW LINE IS     *
3237    0A39    .    .    .    ;            TO BE INSERTED               *
3238    0A39    .    .    .    ;        A,B=NEXT PAGE FIELD ADDR OF LINE *
3239    0A39    .    .    .    ;            TO BE INSERTED
3240    0A39    .    .    .    ;      EXIT :  C,B = A,B(ENTRY)
3241    0A39    .    .    .    ;              D-L DESTROYED
3242    0A39    .    .    .    ;******************************************
3243    0A39    .    .    .    LININA EQU  $
3244    0A39    7B   .    .           MOV  A,E        ;PUT POLLED LINE ADDRESS INT
3245    0A3A    42   .    .           MOV  B,D          ;B,A
3246    0A3B    EB   .    .           XCHG            ;PUT CHAR ADDRESS INTO D,E
3247    0A3C    .    .    .    LININ1 EQU  $
3248    0A3C    6B   .    .           MOV L,E       ;UPDATE PREV LINE PTR
3249    0A3D    62   .    .           MOV H,D       ;IN NEXT LINE
3250    0A3E    23   .    .           INX  H          ;SET ADDRESS TO PREVIOUS
3251    0A3F    23   .    .           INX  H            ;LINE POINTER
3252    0A40    4E   .    .           MOV C,M       ;GET ADDR OF PREV LINE
3253    0A41    77   .    .           MOV M,A       ;STORE ADDR OF NEW LINE
3254    0A42    23   .    .           INX  H
3255    0A43    56   .    .           MOV D,M
3256    0A44    70   .    .           MOV M,B
3257    0A45    .    .    .    ;********************************
3258    0A45    .    .    .    ; UPDATE NEXT/PREVIOUS POINTERS *
3259    0A45    .    .    .    ; IN NEW LINE                   *
3260    0A45    .    .    .    ;********************************
3261    0A45    6F   .    .           MOV L,A       ;GET ADDR OF NEXT LINE FIELD
3262    0A46    7C   .    .           MOV A,H
3263    0A47    60   .    .           MOV H,B
3264    0A48    1D   .    .           DCR E         ;SKIP OVER POINTERS
3265    0A49    73   .    .           MOV M,E       ;STORE NEXT LINE LSB'S
3266    0A4A    23   .    .           INX  H
3267    0A4B    77   .    .           MOV M,A       ;STORE NEXT LINE MSB'S
3268    0A4C    23   .    .           INX  H
3269    0A4D    71   .    .           MOV M,C       ;STORE PREV LINE LSB'S
3270    0A4E    23   .    .           INX  H
3271    0A4F    72   .    .           MOV M,D       ;STORE PREV LINE MSB'S
```

```
=============================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                              PAGE  94
=============================================================================
 3273   0A50    .    .    .   ;*********************************
 3274   0A50    .    .    .   ; SEE IF NEW LINE IS FIRST LINE *
 3275   0A50    .    .    .   ;*********************************
 3276   0A50   79    .    .          MOV A,C        ;GET PREV LINE LSB'S
 3277   0A51   B7    .    .          ORA A          ;SET FLAGS
 3278   0A52   7D    .    .          MOV A,L        ;(PUT LSB OF ADDR IN A-REG)
 3279   0A53   CA   61   0A          JZ  LII200     ;JUMP IF NEW LINE IS
 3280   0A56    .    .    .   ;                         FIRST LINE
 3281   0A56    .    .    .   ;**********************************
 3282   0A56    .    .    .   ; NEW LINE IS NOT FIRST LINE      *
 3283   0A56    .    .    .   ; LINK PREVIOUS LINE TO NEW LINE *
 3284   0A56    .    .    .   ;**********************************
 3285   0A56   D6   04    .          SUI 4          ;GET ADDR OF NEW LINE DATA
 3286   0A58   69    .    .          MOV L,C        ;GET ADDR OF NEXT PAGE FIELD
 3287   0A59   62    .    .          MOV H,D        ;OF PREVIOUS LINE
 3288   0A5A   4F    .    .          MOV  C,A         ;NEW LINE'S LSB TO C
 3289   0A5B   23    .    .          INX  H           ;SET TO MSB PART OF FIELD
 3290   0A5C   CD   95   0F          CALL DISLNK      ;LINK PREV LINE TO NEW LINE
 3291   0A5F   0C    .    .          INR C
 3292   0A60   C9    .    .          RET            ;RETURN
 3293   0A61    .    .    .   ;*************************
 3294   0A61    .    .    .   ; NEW LINE IS FIRST LINE *
 3295   0A61    .    .    .   ;*************************
 3296   0A61    .    .    .   LII200 EQU $
 3297   0A61   D6   03    .          SUI 3          ;GET ADDR OF NEXT PAGE FIELD
 3298   0A63   4F    .    .          MOV C,A        ;PUT LSB INTO C-REGISTER
 3299   0A64   6F    .    .          MOV  L,A         ;SET NEW FIRST LINE POINTER
 3300   0A65   22   9F   FF          SHLD FLINE
 3301   0A68   C9    .    .          RET              ;RETURN
```

```
0A69    .   .   .    ;**********************
0A69    .   .   .    ; LINE FEED PROCESSOR *
0A69    .   .   .    ;**********************
0A69    .   .   .    CONDLF EQU $
0A69    3A  FB  FF          LDA  KBJMPR    ;GET THE STRAP SETTINGS
0A6C    E6  04  .           ANI LINWRP     ;WRAP AROUND ENABLED?
0A6E    C8  .   .           RZ            ;YES - LF NOT REQUIRED
0A6F    .   .   .    LNFEED EQU $
0A6F    21  6C  FF          LXI  H,SPOWL   ;CLEAR SPOW LATCH
0A72    36  FF  .           MVI  M,SPOWOF
0A74    2E  C0  .           MVI  L,CURROW-BASE  ;GET CURSOR ROW
0A76    7E  .   .           MOV A,M
0A77    FE  17  .           CPI MAXROW    ;IS CURSOR IN BOTTOM ROW?
0A79    CA  81  0A          JZ   LNF100    ;YES - ROLL UP THE DISPLAY
0A7C    3C  .   .           INR  A         ;NO - MOVE CURSOR TO NEXT RO
0A7D    77  .   .           MOV  M,A       ;STORE NEW ROW NUMBER
0A7E    32  20  87          STA  IOCRRW    ;SET SCREEN CURSOR
0A81    .   .   .    LNF100 EQU  $
0A81    CC  27  0C          CZ   ROLLUP      ;(ROLL UP IF AT BOTTOM)
0A84    .   .   .    ;
0A84    .   .   .    ;  BUILD FIRST BLOCK OF NEW ROW IF NECESSARY
0A84    .   .   .    ;
0A84    3A  70  FF          LDA  MFLGS     ;GET BLOCK XFR PENDING FLAGS
0A87    E6  40  .           ANI  SENTER/256  ;ENTER PENDING?
0A89    C0  .   .           RNZ            ;YES - DO NOT BUILD NEW ROW
0A8A    3A  64  FF          LDA  IOFLG2    ;NO - GET I/O FLAGS
0A8D    E6  20  .           ANI  XDS2BF    ;DISPLAY TO I/O BUFFER?
0A8F    C0  .   .           RNZ           ;YES - DO NOT BUILD NEW ROW
0A90    .   .   .    ;
0A90    .   .   .    ;  ACQUIRE MEMORY FOR EDIT MODE IF NEEDED
0A90    .   .   .    ;
0A90    3E  FF  .           MVI  A,-1      ;LOCATE BEGINNING OF NEW
0A92    CD  0B  07          CALL RCADR0      ;ROW
0A95    CD  4D  10          CALL CKEDIT    ;CHECK FOR SUFFICIENT FREE
0A98    C4  0A  15          CNZ  FRECNT
0A9B    C9  .   .           RET            ;RETURN
```

```
 3340   0A9C   .    .    .   ;
 3341   0A9C   .    .    .   ; * * * * * * * * * * * * * * * * * * * * * * * * *
 3342   0A9C   .    .    .   ;
 3343   0A9C   .    .    .   ;  LSTLUP - UPDATE "LSTLIN"
 3344   0A9C   .    .    .   ;
 3345   0A9C   .    .    .   ;      ENTRY:  H,L = ADDRESS TO BE STORED
 3346   0A9C   .    .    .   ;
 3347   0A9C   .    .    .   ;      EXIT :  D,E = LSTLIN = H,L(ENTRY)
 3348   0A9C   .    .    .   ;              A,H,L DESTROYED
 3349   0A9C   .    .    .   ;              LSTDCD = 0
 3350   0A9C   .    .    .   ;               PROFLD SET TO INDICATE PROTECTED
 3351   0A9C   .    .    .   ;                FIELD OF FORMAT MODE ENABLED
 3352   0A9C   .    .    .   ;
 3353   0A9C   .    .    .   LSTLUP EQU  $
 3354   0A9C   22   C9   FF         SHLD LSTLIN      ;SET NEW "LSTLIN" VALUE
 3355   0A9F   .    .    .   LSTLU1 EQU  $
 3356   0A9F   EB   .    .          XCHG             ;PUT "LSTLIN" VALUE INTO D,E
 3357   0AA0   .    .    .   LSTLU2 EQU  $
 3358   0AA0   AF   .    .          XRA  A           ;CLEAR LAST DISPLAY CODE
 3359   0AA1   32   C6   FF         STA  LSTDCD
 3360   0AA4   3E   C0   .          MVI  A,STPR      ;INITIALIZE LAST FORMAT
 3361   0AA6   32   C5   FF         STA  LSTFMT        ;CONTROL CODE TO "STPR"
 3362   0AA9   CD   76   19         CALL CHKFMS    ;FORMAT MODE?
 3363   0AAC   C8   .    .          RZ             ;NO - RETURN
 3364   0AAD   3E   FF   .          MVI  A,-1        ;YES - SET PROTECT FLAG TO
 3365   0AAF   32   C2   FF         STA  PROFLD        ;INDICATE PROTECTED FIELD
 3366   0AB2   21   06   07         LXI  H,ZRETRN  ;INITIALIZE FIELD CHECKING
 3367   0AB5   22   86   FF         SHLD CHKRTN        ;ROUTINE
 3368   0AB8   C9   .    .          RET
```

```
3370   0AB9    .    .    .   ;******************
3371   0AB9    .    .    .   ; MEMORY LOCK OFF *
3372   0AB9    .    .    .   ;******************
3373   0AB9    .    .    .   MLKOFO EQU   $
3374   0AB9    3A   6B   FF         LDA   MLKROW      ;GET MEMORY LOCK ROW
3375   0ABC    B7   .    .          ORA   A           ;SET FOR FULL LOCK OUT?
3376   0ABD    C2   C0   10         JNZ   MLKOF       ;NO - CLEAR LOCK OUT ONLY
3377   0AC0    .    .    .   MLKOFF EQU   $           ;YES - TURN OFF MEMORY LOCK
3378   0AC0    21   00   00         LXI   H,0         ;SET MEMORY LOCK ROW AND
3379   0AC3    22   6A   FF         SHLD  MLKFLG        ;FLAG TO ZERO
3380   0AC6    3E   04   .          MVI   A,MEMLOK    ;TURN OFF MEMORY LOCK
3381   0AC8    C3   11   48         JMP   ZCLMD1        ;FLAG
3382   0ACB    .    .    .   ;*****************
3383   0ACB    .    .    .   ; MEMORY LOCK ON *
3384   0ACB    .    .    .   ;*****************
3385   0ACB    .    .    .   MLKON  EQU   $
3386   0ACB    3A   C0   FF         LDA   CURROW      ;GET CURRENT CURSOR ROW
3387   0ACE    B7   .    .          ORA   A           ;SET FOR OVERFLOW INHIBIT?
3388   0ACF    C2   D6   0A         JNZ   MLU005      ;NO - SET MEMORY LOCK ROW
3389   0AD2    CD   4D   10         CALL  CKEDIT      ;EDIT MODE?
3390   0AD5    C0   .    .          RNZ               ;YES - DON'T ALLOW LOCK OUT
3391   0AD6    .    .    .   MLU005 EQU   $           ;NO - SET MEMORY LOCK ROW
3392   0AD6    32   6B   FF         STA   MLKROW
3393   0AD9    .    .    .   MLU010 EQU   $
3394   0AD9    3E   04   .          MVI   A,MEMLOK    ;TURN MEMORY LOCK FLAG
3395   0ADB    06   00   .          MVI   B,0           ;ON AND DON'T BLINK LED
3396   0ADD    21   6A   FF         LXI   H,MLKFLG      ;(CLEAR THE MEMORY LOCK
3397   0AE0    70   .    .          MOV   M,B            ;FLAG)
3398   0AE1    C3   0E   48         JMP   ZSTMD1
```

```
3400   0AE4    .    .    .   ;**********************************
3401   0AE4    .    .    .   ; MLKSCH - LOCATE MEMORY LOCK ROW *
3402   0AE4    .    .    .   ;**********************************
3403   0AE4    .    .    .   ;
3404   0AE4    .    .    .   ;  ENTRY:  DON'T CARE
3405   0AE4    .    .    .   ;
3406   0AE4    .    .    .   ;  EXIT :  Z - MEMORY LOCK ROW NOT FOUND
3407   0AE4    .    .    .   ;            A,C,H,L DESTROYED
3408   0AE4    .    .    .   ;          NZ - MEMORY LOCK ROW FOUND
3409   0AE4    .    .    .   ;            H,L = ADDRESS OF LAST LOCK ROW
3410   0AE4    .    .    .   ;              (POINTS TO LSB OF NEXT LINE
3411   0AE4    .    .    .   ;               POINTER)
3412   0AE4    .    .    .   ;            A,C DESTROYED
3413   0AE4    .    .    .   ;
3414   0AE4    .    .    .   MLKSC0 EQU  $              ;LOCATE FIRST UNLOCKED ROW
3415   0AE4    3A   6B   FF         LDA  MLKROW         ;GET MEMORY LOCK ROW
3416   0AE7    B7   .    .          ORA  A              ;SET FOR PARTIAL SCREEN LOCK
3417   0AE8    2A   CB   FF         LHLD TOPLIN           ;(SET FOR TOP DISPLAY LINE
3418   0AEB    CA   01   0B         JZ   NZEXIT         ;NO - RETURN FOUND (NZ)
3419   0AEE    .    .    .   ;                           YES - LOCATE MEMORY LOCK ROW
3420   0AEE    .    .    .   MLKSCH EQU  $
3421   0AEE    3A   6B   FF         LDA  MLKROW    ;GET MEMORY LOCK ROW
3422   0AF1    B7   .    .          ORA  A              ;SET FOR PARTIAL SCREEN LOCK
3423   0AF2    C8   .    .          RZ                  ;NO - RETURN
3424   0AF3    .    .    .   ;*****************
3425   0AF3    .    .    .   ; SEARCH FOR ROW *
3426   0AF3    .    .    .   ;*****************
3427   0AF3    2A   CB   FF         LHLD TOPLIN         ;GET TOP LINE ADDRESS
3428   0AF6    .    .    .   MLKSC1 EQU  $              ;LOCATE LINE (A-REG)
3429   0AF6    4F   .    .          MOV  C,A            ;PUT LINE NUMBER IN C-REG
3430   0AF7    .    .    .   MLS120 EQU  $
3431   0AF7    CD   6D   19         CALL CHAIN          ;GET ADDRESS OF NEXT LINE
3432   0AFA    B7   .    .          ORA  A              ;DOES NEXT LINE EXIST?
3433   0AFB    C8   .    .          RZ                  ;NO - RETURN FAIL (Z)
3434   0AFC    23   .    .          INX  H              ;YES - SET TO NEXT LINE PTR
3435   0AFD    0D   .    .          DCR  C              ;ALL LINES FOUND?
3436   0AFE    C2   F7   0A         JNZ  MLS120         ;NO - DO NEXT LINE
3437   0B01    .    .    .   ;
3438   0B01    .    .    .   NZEXIT EQU  $
3439   0B01    F6   FF   .          ORI  377Q          ;SET NZ, S
3440   0B03    C9   .    .          RET            ;RETURN(ZERO FLAG FALSE)
```

```
 3442   0B04   .    .    .    ;********************************************
 3443   0B04   .    .    .    ; MLOCK - TURN ON MEMORY LOCK FULL CONDITION *
 3444   0B04   .    .    .    ;********************************************
 3445   0B04   .    .    .    ;
 3446   0B04   .    .    .    ;   ENTRY:  DON'T CARE
 3447   0B04   .    .    .    ;
 3448   0B04   .    .    .    ;   EXIT :  A = 0
 3449   0B04   .    .    .    ;           Z = T
 3450   0B04   .    .    .    ;           MLKTMR = -1 (377B)
 3451   0B04   .    .    .    ;
 3452   0B04   .    .    .    MLOCK0 EQU  $
 3453   0B04   CD   47   06          CALL PTB100     ;RESTORE PROPER DISPLAY PARM
 3454   0B07   .    .    .    MLOCK  EQU  $
 3455   0B07   21   6A   FF          LXI  H,MLKFLG   ;SET H,L TO MEMORY LOCK FLAG
 3456   0B0A   B6   .    .           ORA  M          ;MEMORY ALREADY LOCKED?
 3457   0B0B   C2   19   0B          JNZ  MLK010     ;YES - DON'T SOUND BELL
 3458   0B0E   3E   04   .           MVI  A,MEMLOK   ;NO - FORCE MEMORY LOCK ON
 3459   0B10   06   FF   .           MVI  B,377Q       ;AND BLINKING
 3460   0B12   70   .    .           MOV  M,B        ;SET MEMORY LOCK FLAG
 3461   0B13   CD   0E   48          CALL ZSTMD1
 3462   0B16   .    .    .    MLOCK1 EQU  $          ;SOUND BELL AND RETURN A = 0
 3463   0B16   CD   14   48          CALL ZBELL      ;SOUND THE BELL
 3464   0B19   .    .    .    MLK010 EQU  $
 3465   0B19   AF   .    .           XRA  A          ;SET Z-FLAG
 3466   0B1A   21   9A   FF          LXI  H,NROWS      ;(SET H TO DATA PAGE)
 3467   0B1D   77   .    .           MOV  M,A        ;CLEAR NROWS FOR RCADDR
 3468   0B1E   C9   .    .           RET             ;RETURN (A = 0, Z= T)
 3469   0B1F   00   .    .           NOP             ;NOP FOR PATCH TO "PT772"
 3471   0B20   .    .    .   ;
 3472   0B20   .    .    .   ; * * * * * * * * * * * * * * * * * * * * * * *
 3473   0B20   .    .    .   ;
 3474   0B20   .    .    .   ;  MOVCHR - MOVE CHARACTER STRING
 3475   0B20   .    .    .   ;
 3476   0B20   .    .    .   ;      ENTRY:  H,L = SOURCE POINTER
 3477   0B20   .    .    .   ;              B,C = DESTINATION POINTER
 3478   0B20   .    .    .   ;
 3479   0B20   .    .    .   ;      EXIT :  B,C = NEXT STORAGE LOCATION
 3480   0B20   .    .    .   ;              H,L = END OF SOURCE STRING
 3481   0B20   .    .    .   ;              Z - TERMINATED BY A NULL BYTE
 3482   0B20   .    .    .   ;              NZ - TERMINATED BY AN EOP
 3483   0B20   .    .    .   ;
 3484   0B20   .    .    .   MOVCHR EQU  $
 3485   0B20  7E    .    .          MOV  A,M        ;GET DATA BYTE
 3486   0B21   7    .    .          ORA  A          ;IS IT A NULL?
 3487   0B22   8    .    .          RZ              ;YES - RETURN (Z - TRUE)
 3488   0B23   2    .    .          STAX B          ;NO - STORE THE BYTE
 3489   0B24   3    .    .          INX  H          ;INCREMENT TO NEXT SOURCE BY
 3490   0B25   B    .    .          DCX  B          ;DECREMENT TO NEXT DEST BYTE
 3491   0B26   E   CE    .          CPI  EOP        ;WAS LAST BYTE AN EOP?
 3492   0B28   2   20   0B          JNZ  MOVCHR     ;NO - DO NEXT BYTE
 3493   0B2B   7    .    .          ORA  A          ;YES - SET Z-FALSE
 3494   0B2C   9    .    .          RET             ;RETURN
```

```
3496    0B2D    .    .    .    ;************
3497    0B2D    .    .    .    ; NEXT PAGE *
3498    0B2D    .    .    .    ;************
3499    0B2D    .    .    .    NEXTPG EQU $
3500    0B2D    E    18   .           MVI   A,MAXROW+1   ;COMPUTE NUMBER OF LINES
3501    0B2F    E    6B   .           MVI   L,MLKROW       ;TO ROLL UP
3502    0B31    6    .    .           SUB   M
3503    0B32    CD   45   0B          CALL  NX1100
3504    0B35    .    .    .    NXT040 EQU   $
3505    0B35    3A   6B   FF          LDA   MLKROW      ;SET CURRENT CURSOR POSITION
3506    0B38    32   C0   FF          STA   CURROW        ;TO MEMORY LOCK ROW AND
3507    0B3B    CD   C5   21          CALL  CURPRT       ;LEFT MARGIN
3508    0B3E    CD   76   19          CALL  CHKFMS      ;FORMAT/SOFT KEY DEFINE MODE
3509    0B41    C2   C4   1D          JNZ   FLDSR       ;YES - TAB TO NEXT FIELD
3510    0B44    C9   .    .           RET               ;NO - RETURN
```

```
 3512   0B45   .   .   .    ;
 3513   0B45   .   .   .    ; * * * * * * * * * * * * * * * * * * * * * * * *
 3514   0B45   .   .   .    ;
 3515   0B45   .   .   .    ;  NXTPG1 - ROLL UP N LINES
 3516   0B45   .   .   .    ;
 3517   0B45   .   .   .    ;      ENTRY:  A = NUMBER OF ROWS TO ROLL UP
 3518   0B45   .   .   .    ;              H = BASEH
 3519   0B45   .   .   .    ;
 3520   0B45   .   .   .    ;      EXIT :  C = NUMBER OF LINES ROLLED
 3521   0B45   .   .   .    ;              H,L = NMROLL+
 3522   0B45   .   .   .    ;              A,B,D,E DESTROYED
 3523   0B45   .   .   .    NXT100 EQU $
 3524   0B45   .   .   .    NXTPG1 EQU $
 3525   0B45   4F  .   .           MOV  C,A       ;PUT ROLL PARAMETER IN C-REG
 3526   0B46   2E  82  .           MVI  L,ROLLCT  ;SAVE ROLL PARAMETER
 3527   0B48   71  .   .           MOV  M,C
 3528   0B49   23  .   .           INX  H
 3529   0B4A   .   .   .    NXT110 EQU  $
 3530   0B4A   71  .   .           MOV  M,C
 3531   0B4B   CD  27  0C          CALL ROLLUP    ;ROLL UP SUCCESSFUL?
 3532   0B4E   21  82  FF          LXI  H,ROLLCT    ;(RECALL ROLL COUNT)
 3533   0B51   4E  .   .           MOV  C,M
 3534   0B52   CA  59  0B          JZ   NXT120    ;NO - EXIT
 3535   0B55   0D  .   .           DCR  C         ;ALL LINES DONE?
 3536   0B56   C2  4A  0B          JNZ  NXT110    ;NO - ROLL UP ANOTHER LINE
 3537   0B59   .   .   .    ;                     YES - EXIT (C = 0)
 3538   0B59   .   .   .    ;*********************************************
 3539   0B59   .   .   .    ; TERMINATE ROLL UP - RETURN NUMBER OF LINES *
 3540   0B59   .   .   .    ;   ROLLED                                    *
 3541   0B59   .   .   .    ;*********************************************
 3542   0B59   .   .   .    NXT120 EQU  $
 3543   0B59   23  .   .           INX  H         ;GET NUMBER OF LINES TO BE
 3544   0B5A   7E  .   .           MOV  A,M         ;ROLLED UP
 3545   0B5B   91  .   .           SUB  C         ;COMPUTE ACTUAL NUMBER DONE
 3546   0B5C   4F  .   .           MOV  C,A       ;RETURN VALUE IN C-REGISTER
 3547   0B5D   C9  .   .           RET            ;RETURN
```

```
=========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                    PAGE 103
=========================================================================
 3549    0B5E   .    .    .   ;*****************************************
 3550    0B5E   .    .    .   ; GET ADDRESS OF NEXT                    *
 3551    0B5E   .    .    .   ; RAM BLOCK.                             *
 3552    0B5E   .    .    .   ; ENTRY:                                 *
 3553    0B5E   .    .    .   ; E, BIT 7 = 0, 4K INCREMENTS            *
 3554    0B5E   .    .    .   ;            = 1, 256                    *
 3555    0B5E   .    .    .   ;                                        *
 3556    0B5E   .    .    .   ;     BIT 0 = 0, IN NON-DISPLAY RAM      *
 3557    0B5E   .    .    .   ;           = 1, DISPLAY RAM             *
 3558    0B5E   .    .    .   ;                                        *
 3559    0B5E   .    .    .   ; H = 0 IF FIRST ENTRY OF ROUTINE        *
 3560    0B5E   .    .    .   ;                                        *
 3561    0B5E   .    .    .   ;        CALL NXSBLK                     *
 3562    0B5E   .    .    .   ;                                        *
 3563    0B5E   .    .    .   ; EXIT:                                  *
 3564    0B5E   .    .    .   ;        (H,L) = ADDRESS OF NEXT         *
 3565    0B5E   .    .    .   ;                   BLOCK                *
 3566    0B5E   .    .    .   ;        A = 0 IF END OF MEMORY          *
 3567    0B5E   .    .    .   ;        E SET TO INDICATE APPROP. RAM   *
 3568    0B5E   .    .    .   ;        OTHER REGS. UNCHANGED, FLAGS ARE. *
 3569    0B5E   .    .    .   ;*****************************************
 3570    0B5E   .    .    .   NXSBLK EQU  $
 3571    0B5E   C5   .    .          PUSH B
 3572    0B5F   AF   .    .          XRA  A
 3573    0B60   BC   .    .          CMP  H            ;H = 0?
 3574    0B61   C2   75   0B         JNZ  NXB100       ;NO - ADVANCE TO NEXT BLOCK
 3575    0B64   2A   8D   FF         LHLD BUFBGN       ;IS THERE ANY NON DISPLAY
 3576    0B67   .    .    .   NXB060 EQU  $
 3577    0B67   3A   8C   FF         LDA  BUFEND+1      ;MEMORY?
 3578    0B6A   BC   .    .          CMP  H
 3579    0B6B   D2   83   0B         JNC  NXB200       ;YES, EXIT
 3580    0B6E   2A   AA   FF         LHLD DSPBGN       ;NO, USE DISPLAY MEMORY
 3581    0B71   1C   .    .          INR  E            ;INDICATE DISPLAY MEMORY
 3582    0B72   C3   83   0B         JMP  NXB200       ;EXIT
 3583    0B75   .    .    .   NXB100 EQU  $
 3584    0B75   B3   .    .          ORA  E            ;INCREMENT BY 4K (BIT 7 = 0)
 3585    0B76   01   00   10         LXI  B,10000Q      ;(SET FOR 4K INCREMENT)
 3586    0B79   F2   7E   0B         JP   NXB150       ;YES - COMPUTE NEXT BLOCK AD
 3587    0B7C   06   01   .          MVI  B,256/256    ;NO - INCREMENT BY 256 ONLY
 3588    0B7E   .    .    .   NXB150 EQU  $
 3589    0B7E   09   .    .          DAD  B            ;BUMP POINTER
 3590    0B7F   0F   .    .          RRC               ;TESTING NON-DISPLAY AREA?
 3591    0B80   D2   67   0B         JNC  NXB060       ;YES - CHECK UPPER BOUNDARY
 3592    0B83   .    .    .   NXB200 EQU  $
 3593    0B83   7C   .    .          MOV  A,H          ;IF WE WENT OVER TOP OF
 3594    0B84   .    .    .   ;                          MEMORY H,= 0
 3595    0B84   C1   .    .          POP  B
 3596    0B85   C9   .    .          RET
```

```
==========================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                   PAGE 104
==========================================================================
 3598   0B86    .   .   .   ;*********************************************
 3599   0B86    .   .   .   ; NXTCHR - GET NEXT CHARACTER IN DISPLAY LIST *
 3600   0B86    .   .   .   ;*********************************************
 3601   0B86    .   .   .   ;
 3602   0B86    .   .   .   ;  ENTRY:  D,E = ADDRESS OF CURRENT CHARACTER
 3603   0B86    .   .   .   ;
 3604   0B86    .   .   .   ;  EXIT :  Z = T, CHARACTER IS NOT AN EOL LINK
 3605   0B86    .   .   .   ;                 A = DISPLAY CHARACTER
 3606   0B86    .   .   .   ;                 D,E = ADDRESS OF CHARACTER
 3607   0B86    .   .   .   ;               F, NEXT CHARACTER IS EOL LINK
 3608   0B86    .   .   .   ;                 A DESTROYED
 3609   0B86    .   .   .   ;                 D,E = ADDRESS OF NEXT LINE LINK
 3610   0B86    .   .   .   ;
 3611   0B86    .   .   .   NXTCH0 EQU  $
 3612   0B86    EB  .   .          XCHG             ;PUT POINTER INTO D,E
 3613   0B87    .   .   .   NXTCHR EQU  $
 3614   0B87    1B  .   .          DCX  D           ;GET THE NEXT DISPLAY
 3615   0B88    1A  .   .          LDAX D             ;CHARACTER
 3616   0B89    FE  D0  .          CPI  LNKLIM      ;IS IT A LINK?
 3617   0B8B    DA  99  0B         JC   NCH010      ;NO - EXIT
 3618   0B8E    EB  .   .          XCHG             ;YES - GET NEW ADDRESS
 3619   0B8F    2B  .   .          DCX  H           ;GET LSB OF LINK
 3620   0B90    6E  .   .          MOV  L,M
 3621   0B91    67  .   .          MOV  H,A
 3622   0B92    EB  .   .          XCHG             ;PUT ADDRESS INTO D,E
 3623   0B93    7B  .   .          MOV  A,E         ;PUT LSB INTO A-REGISTER
 3624   0B94    2F  .   .          CMA              ;END OF LINE LINK (LOWER FOU
 3625   0B95    E6  0F  .          ANI  BLKSM         ;BITS NOT ALL ONES)?
 3626   0B97    C0  .   .          RNZ              ;YES - RETURN Z FALSE
 3627   0B98    1A  .   .          LDAX D           ;NO - GET THE DATA BYTE
 3628   0B99    .   .   .   ;
 3629   0B99    .   .   .   NCH010 EQU  $
 3630   0B99    BF  .   .          CMP  A           ;SET Z TRUE
 3631   0B9A    C9  .   .          RET              ;RETURN
```

```
==============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                         PAGE 105
==============================================================================
 3633   0B9B    .   .   .   ;***************************
 3634   0B9B    .   .   .   ; PAROUT - SEND STATUS BITS *
 3635   0B9B    .   .   .   ;***************************
 3636   0B9B    .   .   .   ;
 3637   0B9B    .   .   .   ;  ENTRY:  A = PARITY BITS TO BE SENT
 3638   0B9B    .   .   .   ;
 3639   0B9B    .   .   .   ;  EXIT :  A-E DESTROYED
 3640   0B9B    .   .   .   ;
 3641   0B9B    .   .   .   PAROT4 EQU  $          ;ROTATE DOWN 4 BITS FIRST
 3642   0B9B   0F   .   .          RRC
 3643   0B9C    .   .   .   PAROT3 EQU $
 3644   0B9C   0F   .   .          RRC
 3645   0B9D    .   .   .   PAROT2 EQU $
 3646   0B9D   0F   .   .          RRC
 3647   0B9E    .   .   .   PAROT1 EQU $
 3648   0B9E   0F   .   .          RRC
 3649   0B9F    .   .   .   PAROUT EQU $
 3650   0B9F   E6  0F   .          ANI 17Q       ;GET BITS 0-3
 3651   0BA1   C6  30   .          ADI  ZERO       ;ADD IN ZERO BASE TO FORCE
 3652   0BA3   E5   .   .          PUSH H            ;DISPLAYABLE CHARACTER
 3653   0BA4   CD  CD  FF          CALL ECONTF     ;PERFORM OUTPUT FUNCTION
 3654   0BA7   E1   .   .          POP  H          ;RESTORE H,L
 3655   0BA8   C9   .   .          RET             ;RETURN
```

```
=============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=============================================================================
 3657    0BA9    .   .   .    ;****************
 3658    0BA9    .   .   .    ; PREVIOUS PAGE *
 3659    0BA9    .   .   .    ;****************
 3660    0BA9    .   .   .    PREVPG EQU $
 3661    0BA9    3E  E8  .           MVI  A,-MAXROW-1
 3662    0BAB    2E  6B  .           MVI  L,MLKROW  ;COMPUTE NUMBER OF ROWS TO
 3663    0BAD    86  .   .           ADD  M             ;ROLL DOWN
 3664    0BAE    CD  B6  0B          CALL PRV100
 3665    0BB1    C3  35  0B          JMP NXT040
 3666    0BB4    .   .   .    ;
 3667    0BB4    .   .   .    ;  PRVPG1 - ROLL DOWN FOR CURSOR POSITIONING
 3668    0BB4    .   .   .    ;
 3669    0BB4    .   .   .    ;      ENTRY:  H,L = CURROW+
 3670    0BB4    .   .   .    ;
 3671    0BB4    .   .   .    PRVPG1 EQU $
 3672    0BB4    36  00  .           MVI M,0       ;SET CURRENT ROW TO ZERO
 3673    0BB6    .   .   .    ;
 3674    0BB6    .   .   .    ; * * * * * * * * * * * * * * * * * * * * * * *
 3675    0BB6    .   .   .    ;
 3676    0BB6    .   .   .    ;  PRV100 - ROLL DOWN N LINES
 3677    0BB6    .   .   .    ;
 3678    0BB6    .   .   .    ;      ENTRY:  A = -NUMBER OF LINES TO ROLL DOWN
 3679    0BB6    .   .   .    ;              H = BASEH
 3680    0BB6    .   .   .    ;
 3681    0BB6    .   .   .    ;      EXIT :  A-L DESTROYED
 3682    0BB6    .   .   .    ;
 3683    0BB6    .   .   .    ;
 3684    0BB6    .   .   .    PRV100 EQU $
 3685    0BB6    32  82  FF          STA  ROLLCT  ;SAVE THE ROLL COUNT
 3686    0BB9    .   .   .    PRV110 EQU  $
 3687    0BB9    CD  C5  0B          CALL ROLLDN     ;LINE ROLLED DOWN?
 3688    0BBC    21  82  FF          LXI  H,ROLLCT   ;(SET H TO DATA PAGE)
 3689    0BBF    C8  .   .           RZ              ;NO - RETURN
 3690    0BC0    34  .   .           INR  M          ;ALL LINES DONE?
 3691    0BC1    C2  B9  0B          JNZ  PRV110     ;NO - DO ANOTHER LINE
 3692    0BC4    C9  .   .           RET             ;YES - RETURN
```

```
=========================================================================
ITEM   LOC   OBJECT CODE  SOURCE STATEMENTS                       PAGE 107
=========================================================================
3694   0BC5   .    .    .   ;**************************************
3695   0BC5   .    .    .   ; ROLLDN - ROLL DISPLAY DOWN ONE LINE *
3696   0BC5   .    .    .   ;**************************************
3697   0BC5   .    .    .   ;
3698   0BC5   .    .    .   ;  ENTRY:  DON'T CARE
3699   0BC5   .    .    .   ;
3700   0BC5   .    .    .   ;  EXIT :  NZ - ROLL DOWN SUCCESSFUL
3701   0BC5   .    .    .   ;           Z - ROLL DOWN FAILED
3702   0BC5   .    .    .   ;          ALL REGISTERS DESTROYED
3703   0BC5   .    .    .   ;
3704   0BC5   .    .    .   ROLLDN EQU $
3705   0BC5   CD   EE   0A         CALL MLKSCH
3706   0BC8   CA   EF   0B         JZ   RLD080
3707   0BCB   .    .    .   ;************************
3708   0BCB   .    .    .   ; MEMORY LOCK ROLL DOWN *
3709   0BCB   .    .    .   ;************************
3710   0BCB   EB   .    .          XCHG             ;LAST LOCKED LINE ADDR TO D,
3711   0BCC   2A   CB   FF         LHLD TOPLIN      ;GET TOP LINE ADDRESS
3712   0BCF   23   .    .          INX  H           ;SET ADDRESS TO PREVIOUS LIN
3713   0BD0   23   .    .          INX  H             ;POINTER
3714   0BD1   CD   6D   19         CALL CHAIN       ;GET PREVIOUS LINE'S ADDRESS
3715   0BD4   B7   .    .          ORA  A           ;PREVIOUS LINE EXIST?
3716   0BD5   C8   .    .          RZ               ;NO - RETURN
3717   0BD6   D5   .    .          PUSH D           ;YES - ROLL DOWN THE LINE
3718   0BD7   CD   DA   09         CALL LINDLO      ;DELETE 1ST LINE ABOVE DISP
3719   0BDA   21   A3   FF         LXI  H,TLINO     ;DECREMENT TOP LINE
3720   0BDD   35   .    .          DCR  M             ;NUMBER
3721   0BDE   E1   .    .          POP  H           ;RECALL LAST LOCKED LINE ADD
3722   0BDF   CD   39   0A         CALL LININA      ;ADD LINE BELOW LOCKED LINES
3723   0BE2   3A   6B   FF         LDA  MLKROW      ;GET LOCK ROW NUMBER
3724   0BE5   3D   .    .          DCR  A           ;ADJUST FOR COMPARE
3725   0BE6   21   C7   FF         LXI  H,LSTROW    ;COMPARE TO LAST ROW DONE
3726   0BE9   BE   .    .          CMP M          ;DID IT ROLL DOWN?
3727   0BEA   FA   18   0C         JM   RLD090      ;YES - UPDATE DISPLAY PTRS
3728   0BED   B4   .    .          ORA  H           ;NO - FORCE NZ AND EXIT
3729   0BEF   C9   .    .          RET              ;RETURN
```

```
3731   0BEF    .   .   .    ;*******************
3732   0BEF    .   .   .    ; NORMAL ROLL DOWN *
3733   0BEF    .   .   .    ;*******************
3734   0BEF    .   .   .    RLD080 EQU $
3735   0BEF    3A  6B  FF          LDA  MLKROW     ;GET MEMORY LOCK ROW
3736   0BF2    B7  .   .           ORA  A          ;IS IT ZERO?
3737   0BF3    CA  0A  0C          JZ   RLD085     ;YES - DO NORMAL ROLL DOWN
3738   0BF6    21  C0  FF          LXI  H,CURROW   ;NO - TRY TO ALLOCATE LINES
3739   0BF9    46  .   .           MOV  B,M          ;TO MEMORY LOCK ROW
3740   0BFA    77  .   .           MOV  M,A
3741   0BFB    C5  .   .           PUSH B          ;SAVE CURRENT ROW NUMBER
3742   0BFC    3E  FF  .           MVI  A,-1         ;(SET FOR COLUMN ZERO)
3743   0BFE    CD  0B  07          CALL RCADR0     ;IS MEMORY AVAILABLE?
3744   0C01    C1  .   .           POP  B            ;(RESTORE CURRENT ROW
3745   0C02    78  .   .           MOV  A,B              ;NUMBER)
3746   0C03    32  C0  FF          STA  CURROW
3747   0C06    C0  .   .           RNZ             ;NO - RETURN FAIL
3748   0C07    C3  C5  0B          JMP  ROLLDN     ;YES - RETRY MEMORY LOCK ROL
3749   0C0A    .   .   .    ;
3750   0C0A    .   .   .    ;  DISPLAY NOT LOCKED - DO NORMAL ROLL DOWN
3751   0C0A    .   .   .    ;
3752   0C0A    .   .   .    RLD085 EQU  $
3753   0C0A    2A  CB  FF          LHLD TOPLIN     ;GET TOP LINE ADDRESS
3754   0C0D    23  .   .           INX  H          ;SET TO PREVIOUS LINE
3755   0C0E    23  .   .           INX  H            ;ADDRESS
3756   0C0F    B6  .   .           ORA  M          ;ANY PREVIOUS LINES?
3757   0C10    C8  .   .           RZ              ;NO - DON'T DO ROLL DOWN
3758   0C11    .   .   .    ;                       YES - ROLL ONE LINE DOWN
3759   0C11    .   .   .    ;*****************************
3760   0C11    .   .   .    ; TOP LINE IS NOT FIRST LINE *
3761   0C11    .   .   .    ; ADVANCE POINTERS           *
3762   0C11    .   .   .    ;*****************************
3763   0C11    16  FF  .           MVI D,-1     ;FLAG TO DECREMENT TLINO
3764   0C13    CD  79  0F          CALL TOPUPD  ;UPDATE TOP LINE POINTERS
3765   0C16    2E  C7  .           MVI  L,LSTROW-BASE  ;GET LAST ROW PROCESSED
3766   0C18    .   .   .    RLD090 EQU $
3767   0C18    7E  .   .           MOV A,M
3768   0C19    3C  .   .           INR  A          ;INCREMENT
3769   0C1A    FE  18  .           CPI MAXROW+1
3770   0C1C    C2  05  16          JNZ STOREA    ;NOT ROLL OFF - STORE ROW
3771   0C1F    2A  C9  FF          LHLD LSTLIN     ;GET ADDR OF LAST LINE DONE
3772   0C22    23  .   .           INX  H          ;SET TO PREVIOUS LINE
3773   0C23    23  .   .           INX  H            ;ADDRESS
3774   0C24    C3  53  0C          JMP ROL200
```

```
0C27  .  .  .   ;***********************************
0C27  .  .  .   ; ROLLUP - ROLL UP DISPLAY ONE LINE *
0C27  .  .  .   ;***********************************
0C27  .  .  .   ROLLUP EQU $
0C27  CD EE 0A         CALL MLKSCH
0C2A  CA 66 0C         JZ   ROL080
0C2D  .  .  .   ;**********************
0C2D  .  .  .   ; MEMORY LOCK ROLL-UP *
0C2D  .  .  .   ;**********************
0C2D  7E .  .          MOV A,M       ;IS THERE A NEXT LINE?
0C2E  B7 .  .          ORA A
0C2F  C8 .  .          RZ                ;NO - DON'T DO ROLL UP
0C30  CD DA 09         CALL LINDLO       ;YES - REMOVE FIRST UNLOCKED
0C33  21 A3 FF         LXI  H,TLINO        ;LINE
0C36  34 .  .          INR  M            ;INCREMENT TOP LINE NUMBER
0C37  2A CB FF         LHLD TOPLIN       ;GET TOP DISPLAY LINE ADDRES
0C3A  3A 6B FF         LDA  MLKROW       ;FORCE END-OF-PAGE IF DISPLA
0C3D  F6 20 .          ORI  MAYEOP         ;IS CURRENTLY REFRESHING
0C3F  32 20 87         STA  IOCRRW         ;MEMORY LOCK BOUNDARY ROW
0C42  CD 39 0A         CALL LININA      ;ADD LINE ABOVE DISPLAY
0C45  3A 6B FF         LDA  MLKROW  ;GET LOCK ROW NUMBER
0C48  21 C7 FF         LXI  H,LSTROW  ;GET LAST ROW PROCESSED
0C4B  96 .  .          SUB  M              ;DID IT ROLL UP?
0C4C  FA 74 0C         JM   ROL090         ;YES - UPDATE LINE POINTER
0C4F  C0 .  .          RNZ                 ;NO - RETURN (Z = FALSE)
0C50  77 .  .          MOV  M,A            ;SAME - FORCE LAST ROW = 0
0C51  .  .  .   ROL100 EQU  $
0C51  2E CB .          MVI L,TOPLIN ;SET CURRENT LINE TO TOP LINE
0C53  .  .  .   ROL200 EQU $
0C53  5E .  .          MOV E,M
0C54  .  .  .   ROLUP2 EQU $
0C54  2C .  .          INR L
0C55  56 .  .          MOV D,M
0C56  .  .  .   ;
0C56  .  .  .   ;  ROLUP3 - UPDATE LSTLIN AND CURADR
0C56  .  .  .   ;
0C56  .  .  .   ROLUP3 EQU  $
0C56  EB .  .          XCHG              ;SET LSTLIN TO NEW ROW
0C57  .  .  .   ROLUPC EQU  $
0C57  CD 9C 0A         CALL LSTLUP
0C5A  EB .  .          XCHG              ;PUT NEW ROW ADDRESS INT H,L
0C5B  2B .  .          DCX  H            ;SET TO LSB OF NEXT LINE PTR
0C5C  22 C3 FF         SHLD CURADR       ;SET CURADR TO TOP LINE
0C5F  EB .  .          XCHG              ;RESTORE D,E AND H,L
0C60  AF .  .          XRA  A            ;SET LAST COLUMN PROCESSED
0C61  32 C8 FF         STA  LSTCOL         ;DONE TO ZERO
0C64  B3 .  .          ORA  E            ;SET Z-FLAG FALSE
0C65  C9 .  .          RET               ;RETURN
```

```
==================================================================================
 3825   0C66    .   .   .   ;*****************
 3826   0C66    .   .   .   ; NORMAL ROLL-UP *
 3827   0C66    .   .   .   ;*****************
 3828   0C66    .   .   .   ROL080 EQU $
 3829   0C66    2A  CB  FF          LHLD TOPLIN     ;GET TOP LINE ADDRESS
 3830   0C69    B6  .   .           ORA  M          ;IS TOP LINE LAST LINE?
 3831   0C6A    C8  .   .           RZ              ;YES - RETURN, DON'T ROLL U
 3832   0C6B    16  01  .           MVI  D,1        ;NO - SET D TO INCREMENT
 3833   0C6D    .   .   .   ;                         "TLINO"
 3834   0C6D    3C  .   .           INR  A          ;SET LSB TO NEXT LINE POINTE
 3835   0C6E    .   .   .   ;*****************************
 3836   0C6E    .   .   .   ; TOP LINE IS NOT LAST LINE *
 3837   0C6E    .   .   .   ; ADVANCE POINTERS           *
 3838   0C6E    .   .   .   ;*****************************
 3839   0C6E    .   .   .   ROLUP1 EQU $
 3840   0C6E    CD  79  0F          CALL TOPUPD     ;UPDATE TOP LINE POINTERS
 3841   0C71    21  C7  FF          LXI  H,LSTROW   ;GET LAST ROW # PROCESSED
 3842   0C74    .   .   .   ROL090 EQU  $
 3843   0C74    4E  .   .           MOV C,M
 3844   0C75    0D  .   .           DCR C        ;DECREMENT
 3845   0C76    FA  51  0C          JM  ROL100   ;LINE ROLLED OFF SCREEN
 3846   0C79    71  .   .           MOV M,C      ;STORE UPDATED LSTROW
 3847   0C7A    B4  .   .           ORA H        ;SET Z-FLAG TO FALSE
 3848   0C7B    C9  .   .           RET
 3850    0C7C    .    .    .   ;******************
 3851    0C7C    .    .    .   ; CHAR SET SELECT *
 3852    0C7C    .    .    .   ;******************
 3853    0C7C    .    .    .   SCHRST EQU $
 3854    0C7C   21   58   27          LXI  H,CHRSTB  ;SET FOR CHARACTER SET SELEC
 3855    0C7F   C3   81   04          JMP ESCAP0
 3856    0C82    .    .    .   ;
 3857    0C82    .    .    .   ;  SET NEW ALTERNATE CHARACTER SET
 3858    0C82    .    .    .   ;
 3859    0C82    .    .    .   SCHST1 EQU  $
 3860    0C82   79    .    .          MOV  A,C        ;PUT INPUT CHARACTER IN A-RE
 3861    0C83   E6   0F    .          ANI  17Q        ;EXTRACT CHARACTER SET NUMBE
 3862    0C85   07    .    .          RLC             ;SHIFT TO POSITION FOR
 3863    0C86   07    .    .          RLC               ;ALTERNATE CHARCTER SET
 3864    0C87   07    .    .          RLC
 3865    0C88   07    .    .          RLC
 3866    0C89   32   72   FF          STA  CHRSET  ;STORE CHAR SET SELECT CTL
 3867    0C8C   C9    .    .          RET             ;RETURN
 3869   0C8D    .    .    .    ;
 3870   0C8D    .    .    .    ; * * * * * * * * * * * * * * * * * * * * * * *
 3871   0C8D    .    .    .    ;
 3872   0C8D    .    .    .    ;  SFKYOF - PUT NORMAL DISPLAY ON SCREEN
 3873   0C8D    .    .    .    ;
 3874   0C8D    .    .    .    ;      ENTRY:  DON'T CARE
 3875   0C8D    .    .    .    ;
 3876   0C8D    .    .    .    ;      EXIT :  ALL REGISTERS DESTROYED
 3877   0C8D    .    .    .    ;
 3878   0C8D    .    .    .    SFKYOF EQU  $
 3879   0C8D   CD   8C   19           CALL CHKSFK      ;NORMAL DISPLAY ENABLED?
 3880   0C90   C8    .    .           RZ               ;YES - RETURN
 3881   0C91   3E   F7    .           MVI  A,377Q-DEFSKY  ;NO - SWAP DISPLAY
 3882   0C93   CD   DC   13           CALL CLCMFL      ;CLEAR SOFT KEY MODE FLAG
 3883   0C96   CD   47   10           CALL CKDSPF      ;DISPLAY FUNCTIONS ENABLED?
 3884   0C99   C2   AE   0C           JNZ  SF0010      ;YES - DON'T RESET RANGE TBL
 3885   0C9C   21   64   26           LXI  H,RTABLE  ;NO - RESTORE NORMAL
 3886   0C9F   22   D2   FF           SHLD RNGTA         ;CHARACTER FUNCTION TABLE
 3887   0CA2   C3   AE   0C           JMP  SF0010      ;TURN ON NORMAL DISPLAY
```

```
3889   0CA5   .   .   .   ;*******************************************
3890   0CA5   .   .   .   ; SFKYON - PUT SOFT KEY DISPLAY ON SCREEN *
3891   0CA5   .   .   .   ;*******************************************
3892   0CA5   .   .   .   ;
3893   0CA5   .   .   .   ;  ENTRY:  DON'T CARE
3894   0CA5   .   .   .   ;
3895   0CA5   .   .   .   ;  FXIT :  NZ
3896   0CA5   .   .   .   ;          ALL REGISTERS DESTPOYED
3897   0CA5   .   .   .   ;
3898   0CA5   .   .   .   SFKYON EQU  $
3899   0CA5   CD  8C  19         CALL CHKSFK     ;SOFT KEY DEFINE MODE?
3900   0CA8   C0  .   .          RNZ             ;YES - RETURN
3901   0CA9   3E  08  .          MVI  A,DEFSKY   ;NO - SWAP DISPLAY
3902   0CAB   CD  00  14         CALL STCMFL     ;SET SOFT KEY MODE FLAG
3903   0CAE   .   .   .   ;
3904   0CAE   .   .   .   ;  EXCHANGE DISPLAY
3905   0CAE   .   .   .   ;
3906   0CAE   .   .   .   SF0010 EQU  $
3907   0CAE   CD  69  21         CALL SWAP       ;SWAP DISPLAY PARAMETERS
3908   0CB1   CD  0E  1D         CALL RSTDSP     ;TURN ON THE DISPLAY
3909   0CB4   CD  20  1E         CALL FLDSRX     ;RESCAN LINE TO SET PROPER
3910   0CB7   C3  A4  06         JMP  RCADRA       ;FIELD ATTRIBUTE
3911   0CBA   .   .   .   ;**********************************************
3912   0CBA   .   .   .   ; SFKYDS - DISPLAY CHARACTEP IN SOFT KEY MODE *
3913   0CBA   .   .   .   ;**********************************************
3914   0CBA   .   .   .   ;
3915   0CBA   .   .   .   ;  ENTRY:  DCHAR = CHARACTER TO BE DISPLAYED
3916   0CBA   .   .   .   ;
3917   0CBA   .   .   .   ;  EXIT :  IF CHARACTER FROM KEYBOARD,
3918   0CBA   .   .   .   ;            CHARACTEP IS ADDED TO DISPLAY
3919   0CBA   .   .   .   ;            OTHERWISE, NORMAL DISPLAY IS RESTORED
3920   0CBA   .   .   .   ;
3921   0CBA   .   .   .   SFKYDS EQU  $
3922   0CBA   CD  8C  19         CALL CHKSFK     ;SOFT KEY DEFINE MODE?
3923   0CBD   CA  1A  23         JZ   DSPCHR     ;NO - USE NORMAL POUTINE
3924   0CC0   CD  A6  12         CALL DCXB2D     ;INPUT FROM KEYBOARD?
3925   0CC3   C4  8D  0C         CNZ  SFKYOF     ;NO - SWAP DISPLAY
3926   0CC6   C2  1A  23         JNZ  DSPCHR       ;AND USE NORMAL ROUTINE
3927   0CC9   C3  B6  14         JMP  FDESC1     ;YES - DISPLAY CHARACTER
3928   0CCC   .   .   .   ;                        AND KILL "CURADV" FLAG
```

```
 3930   0CCC    .   .   .    ;**********************
 3931   0CCC    .   .   .    ; SFTRST - SOFT RESET *
 3932   0CCC    .   .   .    ;**********************
 3933   0CCC    .   .   .    SFTRST EQU  $
 3934   0CCC    CD  6E  15          CALL IOBSYC      ;WAIT UNTIL CTU'S FREE
 3935   0CCF    F3  .   .           DI               ;DISABLE INTERRUPTS
 3936   0CD0    3E  01  .           MVI  A,1         ;SET RESET TIMER FOR ONE
 3937   0CD2    32  D0  FF          STA  RSTTMR        ;SECOND ONLY
 3938   0CD5    C3  DA  00          JMP  GO1         ;DO SOFT RESET
```

```
==========================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                     PAGE 115
==========================================================================
3940   0CD8    .    .    .   ;*****************
3941   0CD8    .    .    .   ; SO - SHIFT OUT *
3942   0CD8    .    .    .   ;*****************
3943   0CD8    .    .    .   SHFTOT EQU $
3944   0CD8    CD   8C   19         CALL CHKSFK     ;DEFINE SOFT KEY MODE?
3945   0CDB    C0   .    .          RNZ             ;YES - DON'T SWITCH CHAR SET
3946   0CDC    3A   72   FF         LDA  CHRSET   ;GET CURRENT ALT CHAR SET
3947   0CDF    .    .    .   SHFT1  EQU  $
3948   0CDF    47   .    .          MOV  B,A        ;PUT NEW CHAR SET IN B-REG
3949   0CE0    3E   0B   .          MVI  A,SWCHAR   ;SET CHARACTER SWITCH IN
3950   0CE2    CD   08   48         CALL ZKBCTL       ;KEYBOARD FOR POSSIBLE
3951   0CE5    .    .    .   ;                        FOREIGN MODE ENABLE
3952   0CE5    78   .    .          MOV  A,B        ;RECALL NEW CHARACTER SET
3953   0CE6    .    .    .   SHFT2  EQU  $          ;ENTRY FOR SELF-TEST
3954   0CE6    06   0F   .          MVI  B,17Q      ;SET MASK TO SAVE DISPLAY
3955   0CE8    .    .    .   ;                        ENHANCEMENT BITS
3956   0CE8    C3   E0   21         JMP  DISPC1     ;ADD CODE TO DISPLAY
3957   0CEB    .    .    .   ;****************
3958   0CEB    .    .    .   ; SI - SHIFT IN *
3959   0CEB    .    .    .   ;****************
3960   0CEB    .    .    .   SHFTIN EQU $
3961   0CEB    CD   8C   19         CALL CHKSFK     ;DEFINE SOFT KEY MODE?
3962   0CEE    C0   .    .          RNZ             ;YES - DON'T SWITCH CHAR SET
3963   0CEF    AF   .    .          XRA A         ;SET FOR BASE CHARACTER
3964   0CF0    C3   DF   0C         JMP  SHFT1        ;SET CODE
```

```
 3966   0CF3    .    .    .   ;************************************
 3967   0CF3    .    .    .   ; STATUS - RETURN TERMINAL STATUS *
 3968   0CF3    .    .    .   ;************************************
 3969   0CF3    .    .    .   STATUS EQU  $
 3970   0CF3   01   00   02          LXI  B,SSTAT   ;SET BLOCK TRANSFER FOR
 3971   0CF6   C3   CA   16          JMP  SBLXF0      ;FOR TERMINAL STATUS
 3972   0CF9    .    .    .   ;*************************************
 3973   0CF9    .    .    .   ; STATGO - TRANSMIT TERMINAL STATUS *
 3974   0CF9    .    .    .   ;*************************************
 3975   0CF9    .    .    .   STATGO FQU  $
 3976   0CF9   01   FF   FD          LXI  B,-1-SSTAT
 3977   0CFC   CD   70   10          CALL CLBLXF    ;CLEAR STATUS PENDING FLAG
 3978   0CFF   06   5C    .          MVI  B,ABCKSL  ;SEND <ESC>-<\>
 3979   0D01   CD   BB   17          CALL ESCOUT
 3980   0D04   21   C1   17          LXI  H,XPUTDC  ;SET OUTPUT ROUTINE ADDRESS
 3981   0D07   CD   14   0D          CALL STAPAR    ;OUTPUT STATUS BITS
 3982   0D0A   21   F7   FF          LXI  H,ERRFLG  ;CLEAR DATA COMM ERROR FLAG
 3983   0D0D   7E    .    .          MOV  A,M
 3984   0D0E   E6   FE    .          ANI  377Q-DCMERR
 3985   0D10   77    .    .          MOV  M,A
 3986   0D11   C3   1D   12          JMP  SDTERM    ;SEND TERMINATOR AND RETURN
```

```
=============================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                      PAGE 117
=============================================================================
 3988   0D14   .    .    .  ;*****************************
 3989   0D14   .    .    .  ; STAPAR - OUTPUT STATUS BITS *
 3990   0D14   .    .    .  ;*****************************
 3991   0D14   .    .    .  ;
 3992   0D14   .    .    .  ;  ENTRY:  H,L = ADDRESS OF OUTPUT ROUTINE
 3993   0D14   .    .    .  ;
 3994   0D14   .    .    .  ;  EXIT :  CNTFAD = ADDRESS OF OUTPUT ROUTINE
 3995   0D14   .    .    .  ;          ALL REGISTERS DESTROYED
 3996   0D14   .    .    .  ;
 3997   0D14   .    .    .  STAPAR EQU  $
 3998   0D14   22   CE   FF        SHLD CNTFAD     ;SET OUTPUT ROUTINE VECTOR
 3999   0D17   .    .    .  ;
 4000   0D17   .    .    .  ;  OUTPUT SIZE OF RAM
 4001   0D17   .    .    .  ;
 4002   0D17   3A   AB   FF        LDA  DSPBGN+1   ;COMPUTE NUMBER OF 256-BYTE
 4003   0D1A   2F   .    .         CMA                ;RAM BLOCKS IN DISPLAY
 4004   0D1B   3C   .    .         INR  A             ;AREA
 4005   0D1C   CD   9D   0B        CALL PAROT2     ;TRANSMIT MEMORY SIZE IN K'S
 4006   0D1F   .    .    .  ;
 4007   0D1F   .    .    .  ;  OUTPUT KEYBOARD INTERFACE STRAP SETTINGS
 4008   0D1F   .    .    .  ;
 4009   0D1F   3A   FB   FF        LDA  KBJMPR     ;TRANSMIT STRAPS A-D
 4010   0D22   6F   .    .         MOV  L,A        ;SAVE JUMPER VALUES
 4011   0D23   CD   9F   0B        CALL PAROUT
 4012   0D26   7D   .    .         MOV  A,L        ;RECALL JUMPER VALUES
 4013   0D27   CD   9B   0B        CALL PAROT4     ;TRANSMIT STRAPS E-H
 4014   0D2A   .    .    .  ;
 4015   0D2A   .    .    .  ;  OUTPUT LATCHING KEYS STATUS
 4016   0D2A   .    .    .  ;
 4017   0D2A   3A   F3   FF        LDA  MDFLG2     ;GET TERMINAL MODE FLAGS 2
 4018   0D2D   E6   07   .         ANI  CAPSLK+BLKMDE+AUTOLF  ;EXTRACT BITS
 4019   0D2F   F6   08   .         ORI  10Q        ;ADD BIT 3 TO INDICATE 2645
 4020   0D31   CD   9F   0B        CALL PAROUT     ;SEND LATCHING KEY STATUS
 4021   0D34   .    .    .  ;
 4022   0D34   .    .    .  ;  OUTPUT TERMINAL (2640) TRANSFER PENDING FLAGS
 4023   0D34   .    .    .  ;
 4024   0D34   2A   6F   FF        LHLD MFLGS2     ;GET TERMINAL MODE FLAGS
 4025   0D37   7C   .    .         MOV  A,H        ;MASK FOR SECONDARY STATUS
 4026   0D38   E6   04   .         ANI  SSTAT2/256  ;PENDING BIT
 4027   0D3A   0F   .    .         RRC             ;SHIFT BIT INTO STATUS
 4028   0D3B   0F   .    .         RRC                ;RESPONSE POSITION
 4029   0D3C   0F   .    .         RRC
 4030   0D3D   47   .    .         MOV  B,A
 4031   0D3E   7C   .    .         MOV  A,H        ;GET OTHER DISPLAY RELATED
 4032   0D3F   E6   70   .         ANI  (SENTER+SFCTKY+SCRSEN)/256;XFR BITS
 4033   0D41   B0   .    .         ORA  B          ;ADD IN SECONDARY STATUS
 4034   0D42   CD   9B   0B        CALL PAROT4     ;SEND TRANSFER PENDING BITS
 4036    0D45    .    .    .   ;
 4037    0D45    .    .    .   ;  OUTPUT ERROR CONDITION FLAGS
 4038    0D45    .    .    .   ;
 4039    0D45    06   00   .            MVI   B,0         ;SET FOR NO I/O ERROR
 4040    0D47    3A   4F   FF           LDA   IOCERR      ;GET I/O ERROR FLAG
 4041    0D4A    FE   46   .            CPI   F           ;I/O ERROR OCCURRED?
 4042    0D4C    C2   51   0D           JNZ   STA010      ;NO - GET OTHER ERROR FLAGS
 4043    0D4F    06   08   .            MVI   B,IOERRB    ;YES - SET I/O ERROR BIT
 4044    0D51    .    .    .   STA010 EQU   $
 4045    0D51    00   .    .            NOP               ;INSTR. DELETED TO FIX BUG
 4046    0D52    3A   F7   FF           LDA   ERRFLG      ;GET THE ERROR FLAGS
 4047    0D55    B0   .    .            ORA B           ;MERGE WITH EXISTING BITS
 4048    0D56    CD   9F   0B           CALL PAROUT       ;TRANSMIT ERROR STATUS
 4049    0D59    .    .    .   ;
 4050    0D59    .    .    .   ;  OUTPUT DEVICE TRANSFER PENDING FLAGS
 4051    0D59    .    .    .   ;
 4052    0D59    7C   .    .            MOV   A,H         ;GET TERMINAL MODE 1 FLAGS
 4053    0D5A    07   .    .            RLC               ;PUT I/O DONE FLAG IN C-FLAG
 4054    0D5B    7D   .    .            MOV   A,L         ;GET TERMNAL MODE 2 FLAGS
 4055    0D5C    17   .    .            RAL                 ;ADD IN I/O DONE FLAG
 4056    0D5D    47   .    .            MOV   B,A         ;SAVE TEMPORARY RESULTS
 4057    0D5E    7C   .    .            MOV   A,H         ;RECALL TERMINAL MODE 1 FLAG
 4058    0D5F    0F   .    .            RRC               ;PUT DEVICE STATUS INTO
 4059    0D60    0F   .    .            RRC                 ;C-FLAG
 4060    0D61    0F   .    .            RRC
 4061    0D62    0F   .    .            RRC
 4062    0D63    78   .    .            MOV   A,B           ;RECALL ACCUMULATED BITS
 4063    0D64    17   .    .            RAL                 ;ADD IN DEVICE STATUS
 4064    0D65    C3   9F   0B           JMP   PAROUT      ;SEND DEVICE XFR PENDING BIT
```

```
4066   0D68   .   .   .   ;*******************************************
4067   0D68   .   .   .   ; STCHR1 - SET INITIAL DISPLAY CHARACTER IN *
4068   0D68   .   .   .   ;   NEW DISPLAY BLOCK                       *
4069   0D68   .   .   .   ;*******************************************
4070   0D68   .   .   .   ;
4071   0D68   .   .   .   ;  ENTRY:  H,L = ADDRESS OF FIRST DISPLAY
4072   0D68   .   .   .   ;             IN BLOCK
4073   0D68   .   .   .   ;
4074   0D68   .   .   .   ;  EXIT :  A = 0
4075   0D68   .   .   .   ;          H,L UNCHANGED
4076   0D68   .   .   .   ;
4077   0D68   .   .   .   STCHR1 EQU  $
4078   0D68   3A  F4  FF         LDA  MDFLG1      ;GET SOFT MODE FLAGS
4079   0D6B   E6  80  .          ANI  FORGN       ;FOREIGN MODE ENABLED?
4080   0D6D   3E  CC  .          MVI  A,EOL         ;(SET TO STORE EOL)
4081   0D6F   CA  7A  0D         JZ   STC010      ;NO - STORE EOL ONLY
4082   0D72   2B  .   .          DCX  H           ;YES - STORE EOL AND DISPLAY
4083   0D73   77  .   .          MOV  M,A           ;CONTROL BYTE TO CAUSE
4084   0D74   3A  29  48         LDA  FRSALT        ;FOREIGN CHARACTER SET TO
4085   0D77   F6  80  .          ORI  200Q          ;BE DISPLAYED
4086   0D79   23  .   .          INX  H
4087   0D7A   .   .   .   STC010 EQU  $
4088   0D7A   77  .   .          MOV  M,A         ;STORE FIRST DISPLAY CHAR
4089   0D7B   AF  .   .          XRA  A           ;CLEAR A-REGISTER
4090   0D7C   C9  .   .          RET              ;RETURN
```

```
================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
================================================================================
 4092   0D7D    .    .    .   ;*************************************
 4093   0D7D    .    .    .   ; TEST - PERFORM TERMINAL SELF TEST *
 4094   0D7D    .    .    .   ;*************************************
 4095   0D7D    .    .    .   TEST    EQU  $
 4096   0D7D   CD   4D   10           CALL CKEDIT      ;EDIT MODE ENABLED?
 4097   0D80   C0    .    .           RNZ              ;YES - DON'T DO SELF-TEST
 4098   0D81   3E   08    .           MVI  A,CKIOKY
 4099   0D83   CD   08   48           CALL ZKBCTL      ;I/O CONTROL KEY DOWN ALSO?
 4100   0D86   21   11   28           LXI  H,TSTCTU      ;(SET FOR CTU SELF-TEST)
 4101   0D89   C2   93   15           JNZ  IORMGO      ;YES - DO CTU SELF-TEST
 4102   0D8C    .    .    .   ;                          NO - DO TERMINAL SELF-TEST
 4103   0D8C    .    .    .   ;
 4104   0D8C    .    .    .   ;   PERFORM TERMINAL SELF-TEST
 4105   0D8C    .    .    .   ;
 4106   0D8C    .    .    .   TRMTST  EQU  $
 4107   0D8C   3A   FA   FF           LDA  KBJMP2      ;GET KEYBOARD JUMPERS 2
 4108   0D8F   E6   04    .           ANI  NOTEST      ;SELF-TEST INHIBITED
 4109   0D91   21   51   0F           LXI  H,NOTSMS      ;(SET MESSAGE ADDRESS)
 4110   0D94   C2   D7   1C           JNZ  DSPMS1      ;YES - DISPLAY MSG AND EXIT
 4111   0D97   3A   6E   FF           LDA  DFLGS       ;GET DATA TRANSFER FLAGS
 4112   0D9A   E6   80    .           ANI  XBF2DS      ;DATA FROM I/O BUFFER
 4113   0D9C   C2   D7   1C           JNZ  DSPMS1      ;YES - DON'T DO SELF-TEST
 4114   0D9F   CD   6E   15           CALL IOBSYC      ;WAIT UNTIL CTU'S IDLE
 4115   0DA2   F3    .    .           DI               ;DISABLE INTERRUPTS
 4116   0DA3   3E   05    .           MVI  A,STRTST    ;SET KEYBOARD FOR SELF-TEST
 4117   0DA5   CD   08   48           CALL ZKBCTL        ;START-UP
```

```
=========================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=========================================================================
4119   0DA8    .   .   .   ;*********************
4120   0DA8    .   .   .   ; ROM TEST          *
4121   0DA8    .   .   .   ;                   *
4122   0DA8    .   .   .   ; CALCULATE CHECKSUM *
4123   0DA8    .   .   .   ; FOR EACH 2K ROM    *
4124   0DA8    .   .   .   ;*********************
4125   0DA8   21  00  F8          LXI  H,-NUM2K   ;SET FOR START ADDRESS = 0
4126   0DAB    .   .   .   ;
4127   0DAB    .   .   .   TST010 EQU  $
4128   0DAB   11  00  08          LXI  D,NUM2K    ;INCREMENT START ADDR BY 2K
4129   0DAE   19   .   .          DAD  D
4130   0DAF    .   .   .   ;
4131   0DAF    .   .   .   ; IS CURRENT ADDRESS A ROM?
4132   0DAF    .   .   .   ;
4133   0DAF   7C   .   .          MOV  A,H         ;PUT MSB INTO A-REGISTER
4134   0DB0   FE  B7   .          CPI  134000Q/256-1  ;ADDRESS > 48K?
4135   0DB2   D2  E8  0D          JNC  TST050      ;YES - GO TO NEXT TEST
4136   0DB5   FE  80   .          CPI  100000Q/256  ;IN I/O SPACE?
4137   0DB7   CA  AB  0D          JZ   TST010      ;YES - GO TO NEXT ROM BLOCK
4138   0DBA   FE  88   .          CPI  104000Q/256
4139   0DBC   CA  AB  0D          JZ   TST010      ;YES - GO TO NEXT ROM BLOCK
4140   0DBF   CD  A3  15          CALL IORMG1      ;DOES THE ROM EXIST?
4141   0DC2   CA  CE  0D          JZ   TST020      ;YES - CHECK THE ROM
4142   0DC5   AF   .   .          XRA  A           ;NO - CHECK FOR NO ROM
4143   0DC6   B5   .   .          ORA  L           ;ROM INSTALLED?
4144   0DC7   CA  AB  0D          JZ   TST010      ;NO - GO TO NEXT ROM
4145   0DCA   7C   .   .          MOV  A,H         ;YES - REPORT POSSIBLE
4146   0DCB   C3  D9  0D          JMP  TST030        ;MISPLACED ROM
4147   0DCE    .   .   .   ;*********************
4148   0DCE    .   .   .   ; CALCULATE CHECKSUM *
4149   0DCE    .   .   .   ;*********************
4150   0DCE    .   .   .   TST020 EQU  $
4151   0DCE   2B   .   .          DCX  H           ;RESTORE START ADDRESS
4152   0DCF   16  08   .          MVI  D,NUM2K/256  ;SET TO SUM 2K SPACE
4153   0DD1   CD  81  08          CALL CHKSUM      ;CALCULATE CHECKSUM
4154   0DD4   3C   .   .          INR  A           ;= 377 ?
4155   0DD5    .   .   .   ;*********************************************
4156   0DD5   CA  AB  0D          JZ   TST010      ;YES - DO NEXT ROM BLOCK
4157   0DD8    .   .   .   ;*********************************************
4158   0DD8   AF   .   .          XRA  A           ;NO - REPORT BAD ROM
4159   0DD9    .   .   .   TST030 EQU  $
4160   0DD9   11  37  0F          LXI  D,ROMERR    ;SET ROM ERROR MESSAGE ADDR
4161   0DDC   4F   .   .          MOV  C,A         ;SAVE EXPECTED VALUE
4162   0DDD   46   .   .          MOV  B,M         ;GET VALUE FOUND
4163   0DDE   7C   .   .          MOV  A,H         ;CONVERT ROM ADDRESS TO
4164   0DDF   0F   .   .          RRC                ;ROM NUMBER (0,2,4,...)
4165   0DE0   0F   .   .          RRC
4166   0DE1   6F   .   .          MOV  L,A         ;SET AS ERROR ADDRESS
4167   0DE2   26  00   .          MVI  H,0
4168   0DE4   79   .   .          MOV  A,C         ;RECALL EXPECTED VALUE
4169   0DE5   C3  F3  0E          JMP  TST600      ;REPORT ERROR
```

```
 4171    0DE8    .   .   .   ;*******************************
 4172    0DE8    .   .   .   ; RAM TEST                    *
 4173    0DE8    .   .   .   ;                             *
 4174    0DE8    .   .   .   ; CALCULATE CHECKSUM ON       *
 4175    0DE8    .   .   .   ; EACH 4K BLOCK.              *
 4176    0DE8    .   .   .   ; TEST EACH 256 BYTE SECTION  *
 4177    0DE8    .   .   .   ; RECHECK CHECKSUM.           *
 4178    0DE8    .   .   .   ;*******************************
 4179    0DE8    .   .   .   ;
 4180    0DE8    .   .   .   ;  E = 0
 4181    0DE8    .   .   .   ;
 4182    0DE8    .   .   .   TST050 EQU  $
 4183    0DE8    3E  80  .          MVI  A,CRTOFF   ;TURN OFF VIDEO
 4184    0DEA    32  20  87         STA  IOCRRW
 4185    0DED    21  00  FC         LXI  H,IOBUF    ;SET H,L TO I/O BUFFER #1
 4186    0DF0    CD  FF  10         CALL CLRAL1     ;CLEAR THE I/O BUFFER
 4187    0DF3    44  .   .          MOV  B,H        ;SET B,C = IOBUF2
 4188    0DF4    4D  .   .          MOV  C,L          ;(H,L = IOBUF2)
 4189    0DF5    16  10  .          MVI  D,10000Q/256  ;SET D,E FOR 4K INCREMEN
 4190    0DF7    63  .   .          MOV  H,E        ;SET H TO 0 TO INDICATE STAR
 4191    0DF8    02  .   .          STAX B          ;SET CHECKSUM FOR LAST
 4192    0DF9    .   .   .   ;                        BLOCK TO ZERO
 4193    0DF9    .   .   .   ;*******************************************
 4194    0DF9    .   .   .   ; CALCULATE CHECKSUM FOR EACH RAM BLOCK AND *
 4195    0DF9    .   .   .   ;   STORE CHECKSUM IN "IOBUF2"              *
 4196    0DF9    .   .   .   ;*******************************************
 4197    0DF9    .   .   .   TST060 EQU  $
 4198    0DF9    CD  5E  0B         CALL NXSBLK     ;GET NEXT BLOCK ADDRESS
 4199    0DFC    CD  81  08         CALL CHKSUM     ;COMPUTE CHECKSUM
 4200    0DFF    02  .   .          STAX B          ;STORE CHECKSUM VALUE
 4201    0E00    C2  F9  0D         JNZ  TST060     ;CONTINUE IF NOT LAST BLOCK
```

```
0E03    .   .   .    ;
0E03    .   .   .    ; CHECK EACH 256 BYTE RAM SECTION
0E03    .   .   .    ;
0E03    5D  .   .            MOV   E,L         ;SET E TO ZERO TO INDICATE
0E04    .   .   .    ;                           TESTING OF FAST RAM AREA
0E04    26  91  .            MVI   H,FSTRAM/256  ;START OF FAST RAM (L=0)
0E06    .   .   .    TST090 EQU    $
0E06    .   .   .    ;
0E06    .   .   .    ; TEST THE RAM IN THE FOLLOWING STEPS
0E06    .   .   .    ;
0E06    .   .   .    ; 1. SAVE THE SECTION'S CONTENTS
0E06    01  00  FC           LXI   B,IOBUF     ;I/O BUFFER
0E09    .   .   .    TST100 EQU    $
0E09    7E  .   .            MOV   A,M         ;BYTE TO BE SAVED
0E0A    02  .   .            STAX  B
0E0B    0C  .   .            INR   C           ;SET TO NEXT SAVE ADDRESS
0E0C    .   .   .    ; 2  SET EACH BYTE = MSB .XOR. LSB OF ADDR
0E0C    7D  .   .            MOV   A,L
0E0D    AC  .   .            XRA   H
0E0E    77  .   .            MOV   M,A
0E0F    2C  .   .            INR   L           ;ALL BYTES DONE?
0E10    C2  09  0E           JNZ   TST100      ;NO - DO THE NEXT BYTE
0E13    .   .   .    ; 3. WAIT
0E13    .   .   .    ;       APPROX 2 MS, 5000 CLOCK CYCLES
0E13    .   .   .    TST115 EQU    $
0E13    7F  .   .            MOV   A,A         ;NO OP
0E14    2C  .   .            INR   L
0E15    C2  13  0E           JNZ   TST115
0E18    .   .   .    ; 4. CHECK EACH MEMORY LOCATION
0E18    .   .   .    ;    COMPLEMENT IT
0E18    55  .   .            MOV   D,L         ;D = 0, COUNTER
0E19    2D  .   .            DCR   L           ;L= 377B
0E1A    .   .   .    TST120 EQU    $
0E1A    7D  .   .            MOV   A,L         ;CALCULATE EXPECTED VALUE
0E1B    AC  .   .            XRA   H
0E1C    BE  .   .            CMP   M           ;SAME AS BEFORE?
0E1D    C2  EF  0E           JNZ   TST510      ;NO - REPORT ERROR WITH
0E20    .   .   .    ;                           EXPECTED/FOUND BYTES
0E20    2F  .   .            CMA
0E21    77  .   .            MOV   M,A         ;SET COMPLEMENT
0E22    2D  .   .            DCR   L
0E23    15  .   .            DCR   D           ;DONE WITH THIS SECTION?
0E24    C2  1A  0E           JNZ   TST120      ;NO
0E27    .   .   .    ; 5. WAIT AGAIN
0E27    .   .   .    ;       APPROX 2 MS, 5000 CLOCK CYCLES
0E27    .   .   .    TST125 EQU    $
0E27    7F  .   .            MOV   A,A         ;NO OP
0E28    2D  .   .            DCR   L
0E29    C2  27  0E           JNZ   TST125      ;LOOP FOR 256 TIMES
```

```
4253    0E2C    .   .   .    ; 6. CHECK VALUES. RESTORE ORIGINAL VALUE
4254    0E2C    .   .   .    ;   B,C = IOBUF
4255    0E2C    .   .   .    ;
4256    0E2C    .   .   .    TST130 EQU  $
4257    0E2C    7D  .   .           MOV  A,L
4258    0E2D    AC  .   .           XRA  H
4259    0E2E    2F  .   .           CMA
4260    0E2F    BE  .   .           CMP  M           ;SAME AS BEFORE?
4261    0E30    C2  EF  0E          JNZ  TST510      ;NO - REPORT ERROR WITH
4262    0E33    .   .   .    ;                         EXPECTED/FOUND BYTES
4263    0E33    0A  .   .           LDAX B
4264    0E34    77  .   .           MOV  M,A         ;RESTORE
4265    0E35    03  .   .           INX  B
4266    0E36    2C  .   .           INR  L           ;BLOCK COMPLETED?
4267    0E37    C2  2C  0E          JNZ  TST130      ;NO - DO NEXT BYTE
4268    0E3A    .   .   .    ;***************************
4269    0E3A    .   .   .    ; DONE WITH THIS SECTION.  *
4270    0E3A    .   .   .    ; DO NEXT?                 *
4271    0E3A    .   .   .    ;***************************
4272    0E3A    1C  .   .           INR  E           ;IF E = 0, WE JUST TESTED
4273    0E3B    1D  .   .           DCR  E             ;FAST RAM
4274    0E3C    C2  42  0E          JNZ  TST140
4275    0E3F    63  .   .           MOV  H,E         ;H=0, INDICATE START
4276    0E40    1E  C8  .           MVI  E,200       ;BIT 7 = 1 MEANS 256
4277    0E42    .   .   .    ;                         BYTE INCREMENTS
4278    0E42    .   .   .    ;
4279    0E42    .   .   .    TST140 EQU  $
4280    0E42    CD  5E  0B          CALL NXSBLK      ;GET NEXT BLOCK ADDRESS
4281    0E45    B7  .   .           ORA  A           ;LAST BLOCK DONE?
4282    0E46    C2  06  0E          JNZ  TST090      ;NO, TEST NEXT
```

```
4284   0E49   .   .   .   ;***************************
4285   0E49   .   .   .   ; CHECK ORIGINAL CHECKSUMS *
4286   0E49   .   .   .   ;***************************
4287   0E49   .   .   .   ;
4288   0E49   .   .   .   ;  B,C = IOBUF2
4289   0E49   .   .   .   ;
4290   0E49   59  .   .          MOV  E,C         ;SET E TO ZERO
4291   0E4A   26  FC  .          MVI  H,IOBUF/256  ;SET H TO I/O BUFFER #1
4292   0E4C   CD  FF  10         CALL CLRAL1      ;CLEAR THE I/O BUFFER
4293   0E4F   .   .   .   ;  (H,L) = IOBUF2, TOP HALF OF I/O BUFFER
4294   0E4F   16  10  .          MVI  D,100000/256  ;SET D,E FOR 4K INCREMEN
4295   0E51   7E  .   .          MOV  A,M         ;GET CHECKSUM FOR TOP BLOCK
4296   0E52   73  .   .          MOV  M,E         ;SET STORE BYTE TO ZERO
4297   0E53   F5  .   .          PUSH PSW         ;SAVE TOP BLOCK CHECKSUM
4298   0E54   63  .   .          MOV  H,E         ;SET H TO 0 TO INDICATE STAR
4299   0E55   .   .   .   ;***********************************************
4300   0E55   .   .   .   ; RE-CALCULATE CHECKSUM FOR EACH RAM BLOCK AND *
4301   0E55   .   .   .   ;   COMPARE TO INITIAL STORED VALUE            *
4302   0E55   .   .   .   ;***********************************************
4303   0E55   .   .   .   TST150 EQU  $
4304   0E55   CD  5E  0B         CALL NXSBLK      ;GET NEXT BLOCK ADDRESS
4305   0E58   CD  81  08         CALL CHKSUM      ;COMPUTE CHECKSUM FOR BLOCK
4306   0E5B   6F  .   .          MOV  L,A         ;SAVE COMPUTED VALUE IN L-RE
4307   0E5C   CA  68  0E         JZ   TST160      ;LAST BLOCK - CHECK 1ST VALU
4308   0E5F   0A  .   .          LDAX B           ;RECALL ORIGINAL CHECKSUM
4309   0E60   95  .   .          SUB  L           ;DO CHECKSUMS MATCH?
4310   0E61   6F  .   .          MOV  L,A           ;(SET L TO ZERO IF TRUE)
4311   0E62   CA  55  0E         JZ   TST150      ;YES - GO TO NEXT BLOCK
4312   0E65   C3  ED  0E         JMP  TST500      ;NO - REPORT ERROR
4313   0E68   .   .   .   ;
4314   0E68   .   .   .   TST160 EQU  $
4315   0E68   F1  .   .          POP  PSW         ;RECALL 1ST STORED CHECKSUM
4316   0E69   95  .   .          SUB  L           ;DO CHECKSUMS MATCH?
4317   0E6A   C2  ED  0E         JNZ  TST500      ;NO - REPORT ERROR
```

```
 4319    0E6D    .    .    .   ;***********************
 4320    0E6D    .    .    .   ; DISPLAY TEST PATTERN *
 4321    0E6D    .    .    .   ;***********************
 4322    0E6D    CD   14   48         CALL ZBELL      ;SOUND THE BELL
 4323    0E70    3E   C0   .          MVI  A,300Q     ;SET INITIAL CHARACTER SET
 4324    0E72    .    .    .   TST200 EQU  $
 4325    0E72    D6   10   .          SUI  20Q        ;SET TO NEXT CHARACTER SET
 4326    0E74    F5   .    .          PUSH PSW        ;SAVE CURRENT ENHANCEMENT
 4327    0E75    AF   .    .          XRA  A          ;SET CHARACTER TO NULL
 4328    0E76    32   68   FF         STA  TCHAR
 4329    0E79    .    .    .   TST220 EQU  $
 4330    0E79    CD   B8   21         CALL CRRET    ;DO CR
 4331    0E7C    CD   69   0A         CALL CONDLF   ;DO LF IF WRAPAROUND DISABLED
 4332    0E7F    F1   .    .          POP  PSW        ;RECALL CURRENT ENHANCEMENT
 4333    0E80    F5   .    .          PUSH PSW          ;AND SAVE IT AGAIN
 4334    0E81    CD   E6   0C         CALL SHFT2      ;PUT ENHANCEMENT ON DISPLAY
 4335    0E84    .    .    .   TST240 EQU  $
 4336    0E84    3A   68   FF         LDA  TCHAR      ;GET CURRENT ENHANCEMENT COD
 4337    0E87    32   89   FF         STA  DCHAR    ;STORE CHAR FOR DISPLAY
 4338    0E8A    E6   07   .          ANI 7         ;EVERY 8 CHARS INSERT 2 BLNKS
 4339    0E8C    FE   04   .          CPI  4          ;TIME TO ADD TWO BLANKS?
 4340    0E8E    CC   02   20         CZ   CURAD2     ;YES - ADVANCE CURSOR TWICE
 4341    0E91    CD   1A   23         CALL DSPCHR     ;DISPLAY THE CHARACTER
 4342    0E94    21   68   FF         LXI  H,TCHAR
 4343    0E97    34   .    .          INR  M          ;INCREMENT DISPLAY CHARACTER
 4344    0E98    7E   .    .          MOV  A,M        ;GET NEW CHARACTER
 4345    0E99    FE   40   .          CPI 64
 4346    0E9B    CA   79   0E         JZ   TST220   ;IF 64 THEN NEW LINE
 4347    0E9E    B7   .    .          ORA A         ;ALL CHARACTERS DONE?
 4348    0E9F    F2   84   0E         JP   TST240   ;NO - CONTINUE
 4349    0EA2    CD   96   20         CALL CRLF       ;YES - DOUBLE SPACE BETWEEN
 4350    0EA5    F1   .    .          POP  PSW          ;CHARACTER SETS
 4351    0EA6    FE   80   .          CPI  200Q       ;ALL CHARACTER SETS DONE?
 4352    0EA8    C2   72   0E         JNZ  TST200     ;NO - CONTINUE DISPLAY
```

```
4354   0EAB   .   .   .    ;*******************************
4355   0EAB   .   .   .    ; DISPLAY ENHANCEMENT PATTERN *
4356   0EAB   .   .   .    ;*******************************
4357   0EAB   F5  .   .           PUSH PSW       ;SAVE ENHANCEMENT CODE
4358   0EAC   CD  69  0A          CALL CONDLF    ;DO LF IF WRAPAROUND DISABLE
4359   0EAF   .   .   .    TST420 EQU  $
4360   0EAF   F1  .   .           POP  PSW       ;RECALL CURRENT ENHANCEMENT
4361   0EB0   F5  .   .           PUSH PSW       ;SAVE ENHANCEMENT AGAIN
4362   0EB1   D6  40  .           SUI  100Q      ;COMPUTE ASCII DISPLAY CODE
4363   0EB3   CD  14  23          CALL DSPTST    ;DISPLAY THE CHARACTER
4364   0EB6   F1  .   .           POP  PSW       ;RECALL CURRENT ENHANCEMENT
4365   0EB7   3C  .   .           INR  A         ;INCREMENT ENHANCEMENT
4366   0EB8   FE  90  .           CPI  220Q      ;LAST ENHANCEMENT DONE?
4367   0EBA   CA  C4  0E          JZ   TST440    ;YES - DISPLAY STATUS
4368   0EBD   F5  .   .           PUSH PSW       ;NO - SAVE ENHANCEMENT CODE
4369   0EBE   CD  DE  21          CALL DISPC0    ;ADD ENHANCEMENT TO DISPLAY
4370   0EC1   C3  AF  0E          JMP  TST420    ;DISPLAY ASCII DISPLAY CODE
4371   0EC4   .   .   .    ;
4372   0EC4   .   .   .    TST440 EQU  $
4373   0EC4   AF  .   .           XRA  A
4374   0EC5   CD  DE  21          CALL DISPC0    ;RETURN TO NORMAL VIDEO
4375   0EC8   CD  02  20          CALL CURAD2    ;ADVANCE CURSOR TWICE
4376   0ECB   .   .   .    ;***************************
4377   0ECB   .   .   .    ; DISPLAY TERMINAL STATUS *
4378   0ECB   .   .   .    ;***************************
4379   0ECB   21  F7  FF          LXI  H,ERRFLG  ;SET ERROR FLAG TO
4380   0ECE   7E  .   .           MOV  A,M          ;SELF-TEST SUCCESSFUL
4381   0ECF   F6  02  .           ORI  TESTOK
4382   0ED1   77  .   .           MOV  M,A
4383   0ED2   21  14  23          LXI  H,DSPTST  ;SET H,L TO OUTPUT ROUTINE
4384   0ED5   CD  14  0D          CALL STAPAR    ;DISPLAY TERMINAL STATUS
4385   0ED8   CD  05  20          CALL CURADV    ;PUT SPACE BETWEEN STATUS
4386   0EDB   CD  29  26          CALL STA2G2
4387   0EDE   CD  96  20          CALL CRLF
4388   0EE1   CD  96  20          CALL CRLF
4389   0EE4   .   .   .    ;***********************
4390   0EE4   .   .   .    ; TERMINATE SELF-TEST *
4391   0EE4   .   .   .    ;***********************
4392   0EE4   3E  06  .           MVI  A,ENDTST  ;RESTORE KEYBOARD LED'S
4393   0EE6   CD  08  48          CALL ZKBCTL
4394   0EE9   FB  .   .           EI             ;RE-ENABLE INTERRUPTS
4395   0EEA   C3  23  20          JMP  CRADV1    ;RESET CURSOR ADVANCE FLAG
```

```
 4397    0EED    .    .    .   TST500 EQU  $           ;REPORT RAM ERROR
 4398    0EED    AF   .    .          XRA  A           ;SET Z TRUE FOR ADDRESS ONLY
 4399    0EEE    6F   .    .          MOV  L,A         ;FORCE L-REGISTER TO BE ZERO
 4400    0EEF    .    .    .   TST510 EQU  $
 4401    0EEF    46   .    .          MOV  B,M         ;PUT VALUE FOUND INTO B-REG
 4402    0EF0    11   3B   0F         LXI  D,RAMERR    ;SET D,E TO ERROR MESSAGE
 4403    0EF3    .    .    .   ;****************************
 4404    0EF3    .    .    .   ; REPORT ROM/RAM TEST ERROR *
 4405    0EF3    .    .    .   ;****************************
 4406    0EF3    .    .    .   ;
 4407    0EF3    .    .    .   ;  ENTRY:  D,E = ADDRESS AT WHICH ERROR OCCURRED
 4408    0EF3    .    .    .   ;          H,L = ERROR MESSAGE ADDRESS
 4409    0EF3    .    .    .   ;          Z - DISPLAY ERROR ADDRESS ONLY
 4410    0EF3    .    .    .   ;          NZ - DISPLAY PARAMETERS ALSO
 4411    0EF3    .    .    .   ;            A = EXPECTED VALUE
 4412    0EF3    .    .    .   ;            (H,L) = VALUE FOUND
 4413    0EF3    .    .    .   ;
 4414    0EF3    .    .    .   TST600 EQU  $
 4415    0EF3    EB   .    .          XCHG             ;(H,L) = MESSAGE ADDRESS
 4416    0EF4    .    .    .   ;                        (D,E) = ERROR ADDRESS
 4417    0EF4    E5   .    .          PUSH H           ;SAVE THE MESSAGE ADDRESS
 4418    0EF5    21   50   0F         LXI  H,ERREOP    ;SET EOP FOR SHORT MESSAGE
 4419    0EF8    22   EB   FF         SHLD MSGPT4
 4420    0EFB    CA   0E   0F         JZ   TST610      ;Z - SHOW ADDRESS ONLY
 4421    0EFE    21   00   FD         LXI  H,IOBUF2    ;SET BUFFER ADDRESS
 4422    0F01    22   EB   FF         SHLD MSGPT4
 4423    0F04    C5   .    .          PUSH B           ;SAVE VALUE FOUND
 4424    0F05    CD   02   08         CALL BINOCT      ;CONVERT BINARY TO OCTAL
 4425    0F08    F1   .    .          POP  PSW         ;RECALL VALUE FOUND
 4426    0F09    CD   02   08         CALL BINOCT      ;CONVERT BINARY TO OCTAL
 4427    0F0C    36   CE   .          MVI  M,EOP       ;TERMINATE WITH "EOP"
 4428    0F0E    .    .    .   TST610 EQU  $
 4429    0F0E    21   10   FD         LXI  H,IOBUF2+16  ;CONVERT FAILURE ADDRESS
 4430    0F11    22   ED   FF         SHLD MSGPT3
 4431    0F14    CD   2E   08         CALL BN2DEC      ;CONVERT TO DECIMAL ASCII
 4432    0F17    21   42   0F         LXI  H,RXMERR    ;SET REST OF LITERAL
 4433    0F1A    22   EF   FF         SHLD MSGPT2
 4434    0F1D    3A   6E   FF         LDA  DFLGS       ;GET DATA TRANSFER FLAGS
 4435    0F20    E6   01   .          ANI  SDACOM      ;TEST FROM DATA COMM?
 4436    0F22    E1   .    .          POP  H             ;(RECALL MESSAGE ADDRESS)
 4437    0F23    CA   54   12         JZ   HANGU0      ;NO - SHOW MESSAGE AND HANG
 4438    0F26    C7   .    .          RST  ;0           YES - RESET THE TERMINAL
```

```
=============================================================================
 ITEM     LOC    OBJECT CODE  SOURCE STATEMENTS                        PAGE 129
=============================================================================
 4440    0F27    .    .    .   ;*****************
 4441    0F27    .    .    .   ; MESSAGE STORAGE *
 4442    0F27    .    .    .   ;*****************
 4443    0F27    .    .    .   BUFMSG EQU   $
 4444    0F27    42   55   46         DB    'BUFFER OVERFLOW',EOP
 4445    0F37    .    .    .   ;
 4446    0F37    .    .    .   ROMERR EQU   $
 4447    0F37    52   4F   4D         DB    'ROM',0
 4448    0F3B    .    .    .   ;
 4449    0F3B    .    .    .   RAMERR EQU   $
 4450    0F3B    52   41   4D         DB    'RAM',0
 4451    0F3F    .    .    .   ;
 4452    0F3F    .    .    .   INERMS EQU   $
 4453    0F3F    49   2F   4F         DB    'I/O'
 4454    0F42    .    .    .   ;
 4455    0F42    .    .    .   RXMERR EQU   $
 4456    0F42    20   45   52         DB    ' ERROR ',0
 4457    0F4A    .    .    .   ;
 4458    0F4A    .    .    .   LDRMSG EQU   $
 4459    0F4A    4C   4F   41         DB    'LOADER'
 4460    0F50    .    .    .   ERREOP EQU   $
 4461    0F50    CE   .    .          DB    EOP
 4462    0F51    .    .    .   ;
 4463    0F51    .    .    .   NOTSMS EQU   $
 4464    0F51    4E   4F   20         DB    'NO TEST',EOP
 4465    0F59    .    .    .   ;
 4466    0F59    .    .    .   NODRVR EQU   $
 4467    0F59    4E   4F   20         DB    'NO DEVICE DRIVER',EOP
 4468    0F6A    .    .    .   ;
 4469    0F6A    .    .    .   TRMRDY EQU   $
 4470    0F6A    54   45   52         DB    'TERMINAL READY',EOP
```

```
4472   0F79   .   .   .   ;**********************************
4473   0F79   .   .   .   ; TOPUPD - UPDATE TOP LINE POINTERS *
4474   0F79   .   .   .   ;**********************************
4475   0F79   .   .   .   TOPUPD EQU $
4476   0F79   23  .   .          INX  H          ;PUT THE MSB INTO THE
4477   0F7A   46  .   .          MOV  B,M          ;B-REGISTER
4478   0F7B   4F  .   .          MOV  C,A        ;SAVE TOP LINE'S LSB IN C-RE
4479   0F7C   21  A3  FF         LXI  H,TLINO    ;UPDATE TOP LINE NUMBER
4480   0F7F   7A  .   .          MOV A,D
4481   0F80   B7  .   .          ORA A         ;IS TLINO TO BE RESET?
4482   0F81   CA  85  0F         JZ  TOP100    ;YES
4483   0F84   86  .   .          ADD M         ;NO - INCREMENT OR DECREMENT
4484   0F85   .   .   .   TOP100 EQU $
4485   0F85   77  .   .          MOV M,A       ;STORE UPDATED TLINO
4486   0F86   .   .   .   TOPUP1 EQU  $
4487   0F86   60  .   .          MOV  H,B        ;SET NEW TOP LINE POINTER
4488   0F87   69  .   .          MOV  L,C
4489   0F88   22  CB  FF         SHLD TOPLIN
4490   0F8B   3A  F8  FF         LDA  CMFLGS     ;GET COMMON FLAGS
4491   0F8E   E6  08  .          ANI  DEFSKY     ;SOFT KEY DEFINE MODE?
4492   0F90   C0  .   .          RNZ             ;YES - DON'T CHANGE SCREEN
4493   0F91   21  FF  FF         LXI  H,DISPST+1  ;GET DISPLAY START ADDRESS
4494   0F94   0B  .   .          DCX  B          ;SET TO FIRST CHAR ADDRESS
4495   0F95   .   .   .   ;************************************
4496   0F95   .   .   .   ; DISLNK - STORE LINK IN DISPLAY AREA *
4497   0F95   .   .   .   ;************************************
4498   0F95   .   .   .   ;
4499   0F95   .   .   .   ;  ENTRY:  B,C = LINK TO BE STORED
4500   0F95   .   .   .   ;          H,L = STORE ADDRESS FOR MSB PART
4501   0F95   .   .   .   ;
4502   0F95   .   .   .   ;  EXIT :  H,L = LSB OF STORE ADDRESS
4503   0F95   .   .   .   ;          A DESTROYED
4504   0F95   .   .   .   ;          INTERRUPTS ENABLED
4505   0F95   .   .   .   ;
4506   0F95   .   .   .   DISLNK EQU  $
4507   0F95   3E  60  .          MVI  A,DMAOFF   ;SET TO TURN OFF THE DMA
4508   0F97   F3  .   .          DI              ;DISABLE INTERRUPTS
4509   0F98   32  20  87         STA  IOCRRW     ;TURN OFF DMA
4510   0F9B   70  .   .          MOV  M,B        ;STORE LINK'S MSB
4511   0F9C   2B  .   .          DCX  H
4512   0F9D   71  .   .          MOV  M,C        ;STORE LINK'S MSB
4513   0F9E   .   .   .   DISLN1 EQU  $        ;SET CURSOR ROW POSITION
4514   0F9E   3A  C0  FF         LDA  CURROW     ;TURN DMA BACK ON WITH
4515   0FA1   .   .   .   DISLN2 EQU  $
4516   0FA1   32  20  87         STA  IOCRRW       ;CURRENT CURSOR ROW ADDRES
4517   0FA4   .   .   .   DISLN3 EQU  $
4518   0FA4   FB  .   .          EI              ;RE-ENABLE INTERRUPTS
4519   0FA5   .   .   .   DISLN4 EQU  $        ;RE-ENABLE RESET KEY
4520   0FA5   3E  02  .          MVI  A,RSTON
4521   0FA7   32  80  83         STA  IOKBCO
4522   0FAA   C9  .   .          RET             ;RETURN
```

```
4524   0FAB   .    .    .    ;********************************
4525   0FAB   .    .    .    ; TYPSET - SET TYPE DEFINITION *
4526   0FAB   .    .    .    ;********************************
4527   0FAB   .    .    .    TYPSET EQU  $
4528   0FAB   CD   76   19          CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
4529   0FAE   C0   .    .           RNZ             ;YES - DO SET TYPE
4530   0FAF   3A   89   FF          LDA  DCHAR      ;NO - COMPUTE TYPE DEFINITIO
4531   0FB2   C6   8F   .           ADI  ALPHA-ZERO-6  ;CHARACTER
4532   0FB4   C3   E0   21          JMP  DISPC1     ;ADD CHARACTER TO DISPLAY
4533   0FB7   .    .    .    ;*************************************
4534   0FB7   .    .    .    ; SFKCHK - SOFT KEY ATTRIBUTE CHECK *
4535   0FB7   .    .    .    ;*************************************
4536   0FB7   .    .    .    SFKCHK EQU  $
4537   0FB7   E6   DF   .           ANI  377Q-40Q   ;FORCE INPUT TO UPPER CASE
4538   0FB9   2A   C3   FF          LHLD CURADR     ;RECALL CHARACTER ADDRESS
4539   0FBC   77   .    .           MOV  M,A        ;STORE UPPER CASE VALUE
4540   0FBD   FE   4E   .           CPI  N          ;NORMAL ATTRIBUTE SET?
4541   0FBF   C8   .    .           RZ              ;YES - RETURN SUCCESSFUL
4542   0FC0   FE   4C   .           CPI  L          ;LOCAL ATTRIBUTE SET?
4543   0FC2   C8   .    .           RZ              ;YES - RETURN SUCCESSFUL
4544   0FC3   FE   54   .           CPI  T          ;TRANSMIT ONLY SET?
4545   0FC5   C8   .    .           RZ              ;YES - RETURN SUCCESSFUL
4546   0FC6   70   .    .           MOV  M,B        ;NO - RESTORE ORIGINAL
4547   0FC7   C9   .    .           RET               ;ATTRIBUTE AND RETURN NZ
```

```
=================================================================================
 ITEM     LOC    OBJECT CODE  SOURCE STATEMENTS
=================================================================================
 4549    0FC8    .     .     .     ;
 4550    0FC8    .     .     .     ; * * * * * * * * * * * * * * * * * * * * * * * *
 4551    0FC8    .     .     .     ;
 4552    0FC8    .     .     .     ;   XMS2DS - TRANSFER MESSAGF TO NORMAL DISPLAY
 4553    0FC8    .     .     .     ;
 4554    0FC8    .     .     .     ;        ENTRY:  H,L = POINTER TO MESSAGE
 4555    0FC8    .     .     .     ;
 4556    0FC8    .     .     .     ;        EXIT :  A-L DESTROYED
 4557    0FC8    .     .     .     ;                Z - TERMINATED BY A NULL BYTE
 4558    0FC8    .     .     .     ;                NZ - TERMINATED BY AN EOP
 4559    0FC8    .     .     .     ;
 4560    0FC8    .     .     .     XMD000 EQU  $
 4561    0FC8    CD    14    23           CALL DSPTST     ;DISPLAY ASCII CHARACTER AND
 4562    0FCB    .     .     .     ;                        ADVANCE CURSOR
 4563    0FCB    .     .     .     XMD010 EQU  $
 4564    0FCB    E1    .     .            POP  H          ;RESTORE H AND L
 4565    0FCC    23    .     .            INX  H          ;MOVE TO NEXT BYTE
 4566    0FCD    .     .     .     ;                      PROCESS THE NEXT BYTE
 4567    0FCD    .     .     .     XMS2DS EQU  $
 4568    0FCD    7E    .     .            MOV  A,M        ;SET THE SOURCE BYTE
 4569    0FCE    B7    .     .            ORA  A          ;IS IT A NULL BYTE?
 4570    0FCF    C8    .     .            RZ              ;YES - RETURN (Z - TRUE)
 4571    0FD0    FE    CE    .            CPI  EOP        ;IS IT END OF PAGE FLAG?
 4572    0FD2    CA    ED    0F           JZ   XMD030     ;YES - EXIT
 4573    0FD5    E5    .     .            PUSH H          ;NO - SAVE H,L
 4574    0FD6    FE    CC    .            CPI  EOL        ;IS IT AN END OF LINE?
 4575    0FD8    CA    E7    0F           JZ   XMD020     ;YES - START A NEW LINE
 4576    0FDB    B7    .     .            ORA  A          ;IS CHARACTER ASCII?
 4577    0FDC    F2    C8    0F           JP   XMD000     ;YES - DISPLAY IT
 4578    0FDF    06    00    .            MVI  B,0        ;NO - FORCE ENHANCEMENT CODE
 4579    0FE1    CD    E2    21           CALL DISPC2       ;TO BE STORED AS IS
 4580    0FE4    C3    CB    0F           JMP  XMD010     ;GO TO NEXT BYTE
 4581    0FE7    .     .     .     ;
 4582    0FE7    .     .     .     ;  EOL CODE - TERMINATE THE LINE
 4583    0FE7    .     .     .     ;
 4584    0FE7    .     .     .     XMD020 EQU  $
 4585    0FE7    CD    96    20           CALL CRLF       ;PERFORM RETURN AND LINE FEE
 4586    0FEA    C3    CB    0F           JMP  XMD010     ;DO NEXT BYTE
 4587    0FED    .     .     .     ;
 4588    0FED    .     .     .     ;  EOP CODE - TERMINATE LINE AND EXIT
 4589    0FED    .     .     .     ;
 4590    0FED    .     .     .     XMD030 EQU  $
 4591    0FED    CD    96    20           CALL CRLF       ;PUT CURSOR IN NEXT LINE
 4592    0FF0    B4    .     .            ORA  H          ;SET Z FALSE
 4593    0FF1    C9    .     .            RET             ;RETURN TEMINATED BY EOP
```

```
0FF2  .  .  .   ;*********************************************
0FF2  .  .  .   ; CARRET - PERFORM DISPLAY FUNCTIONS RETURN *
0FF2  .  .  .   ;*********************************************
0FF2  .  .  .   CARRET EQU  $
0FF2  CD 8C 19         CALL CHKSFK     ;SOFT KEY DEFINE MODE?
0FF5  CA 0B 10         JZ   CAR010     ;NO - DO NORMAL PROCESSING
0FF8  .  .  .   ;**************************
0FF8  .  .  .   ;  R O M   B R E A K   2  *
0FF8  .  .  .   ;**************************
0FF8  C3 02 10         JMP  ZBRK2C     ;GO TO NEXT ROM BLOCK
0FFB  .  .  .          ORG  ZBRK1+4000Q
1000  .  .  .   ZBRK2  EQU  $
1000  50 .  .          DB   VERSN      ;ROM PRESENT FLAGS
1001  10 .  .          DB   ZBRK2/256
1002  .  .  .   ZBRK2C EQU  $
1002  .  .  .   ;************************************************
1002  CD A6 12         CALL DCXB2D     ;DATA FROM KEYBOARD?
1005  CA 1A 23         JZ   DSPCHR     ;YES - DISPLAY RETURN CODE
1008  CD 8D 0C         CALL SFKYOF     ;NO - RESTORE NORMAL DISPLAY
100B  .  .  .   CAR010 EQU  $
100B  21 96 20         LXI  H,CRLF     ;SET NORMAL ROUTINE EXIT
100E  C3 1D 23         JMP  DSPCH0     ;DISPLAY THE CHARACTER
```

```
=====================================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS
=====================================================================================
 4618   1011    .    .    .    ;**********************************************
 4619   1011    .    .    .    ; CHKLIM - CHECK PARAMETER BOUNDARY CONDITIONS *
 4620   1011    .    .    .    ;**********************************************
 4621   1011    .    .    .    ;
 4622   1011    .    .    .    ;  ENTRY:  B = CURRENT VALUE
 4623   1011    .    .    .    ;          C = MAXIMUM ALLOWABLE VALUE
 4624   1011    .    .    .    ;          D,E = ADDRESS OF PARAMETER TO BE SET
 4625   1011    .    .    .    ;          IODATA = PARAMETER VALUE (2 BYTES)
 4626   1011    .    .    .    ;          IOPSGN = -1, NEGATIVE ADJUSTMENT
 4627   1011    .    .    .    ;                 =  0, ABSOLUTE VALUE
 4628   1011    .    .    .    ;                 = +1, POSITIVE ADJUSTMENT
 4629   1011    .    .    .    ;
 4630   1011    .    .    .    ;  EXIT :  NEW VALUE IN WORD ADDRESSED BY D,E
 4631   1011    .    .    .    ;          A,C,H,L DESTROYED
 4632   1011    .    .    .    ;
 4633   1011    .    .    .    ;  THIS ROUTINE SET THE NEW VALUE BY EITHER
 4634   1011    .    .    .    ;  OR ABSOLUTE ADJUST WITHING THE LIMITS OF
 4635   1011    .    .    .    ;  ZERO AND THE MAXIMUM ALLOWABLE AS SPECIFIED
 4636   1011    .    .    .    ;  THE C-REGISTER ON ENTRY
 4637   1011    .    .    .    ;
 4638   1011    .    .    .    ;  THE LARGEST MAXIMUM VALUE IS 255
 4639   1011    .    .    .    ;
 4640   1011    .    .    .    CHKLIO EQU  $
 4641   1011    3A   DD   FF          LDA  IOCSGN     ;SET PARAMETER SIGN TO
 4642   1014    32   DC   FF          STA  IOPSGN       ;INPUT SIGN
 4643   1017    .    .    .    CHKLIM EQU  $
 4644   1017    3A   DF   FF          LDA  IODATA+1   ;GET MSB OF INPUT VALUE
 4645   101A    B7   .    .           ORA  A          ;MAXIMUM EXCEEDED?
 4646   101B    3A   DC   FF          LDA  IOPSGN       ;(GET PARAMETER SIGN)
 4647   101E    CA   28   10          JZ   CHK050     ;NO - CONTINUE EVALUATION
 4648   1021    87   .    .           ADD  A          ;NEGATIVE ADJUSTMENT?
 4649   1022    F2   37   10          JP   CHK070     ;NO - SET TO MAXIMUM VALUE
 4650   1025    .    .    .    ;
 4651   1025    .    .    .    ;  DEFAULT TO MINIMUM VALUE (0)
 4652   1025    .    .    .    ;
 4653   1025    .    .    .    CHK010 EQU $         ;SET TO ZERO
 4654   1025    AF   .    .           XRA A
 4655   1026    12   .    .           STAX D          ;STORE NEW VALUE
 4656   1027    C9   .    .           RET             ;RETURN
```

```
=================================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                            PAGE 135
=================================================================================
4658   1028   .   .   .    ;
4659   1028   .   .   .    ;  PARAMETER < 256, EVALUATE FOR RELATIVE AMOUNT
4660   1028   .   .   .    ;
4661   1028   .   .   .    CHK050 EQU $
4662   1028   21  DE  FF          LXI  H,IODATA  ;SET H,L TO GET INPUT VALUE
4663   102B   87  .   .           ADD  A           ;RELATIVE POSITIONING?
4664   102C   78  .   .           MOV A,B      ;(LOAD CURRENT VALUE)
4665   102D   FA  41  10          JM  CHK160    ;MINUS - SUBTRACT INPUT
4666   1030   C2  3A  10          JNZ CHK150    ;PLUS - ADD INPUT
4667   1033   7E  .   .           MOV A,M      ;NONE - ABSOLUTE ASSIGNMENT
4668   1034   .   .   .    ;
4669   1034   .   .   .    ;  CHECK UPPER LIMIT + 1
4670   1034   .   .   .    ;
4671   1034   .   .   .    CHK060 EQU  $
4672   1034   12  .   .           STAX D          ;STORE ASSIGNED VALUE
4673   1035   B9  .   .           CMP  C          ;MAXIMUM EXCEEDED?
4674   1036   D8  .   .           RC              ;NO - RETURN
4675   1037   .   .   .    CHK070 EQU  $          ;YES - USE MAXIMUM VALUE
4676   1037   79  .   .           MOV  A,C
4677   1038   .   .   .    ;************************
4678   1038   .   .   .    ; STORE PARAMETER VALUE *
4679   1038   .   .   .    ;************************
4680   1038   .   .   .    CHK100 EQU  $
4681   1038   12  .   .           STAX D          ;STORE PARAMETER VALUE
4682   1039   C9  .   .           RET             ;RETURN
4683   103A   .   .   .    ;
4684   103A   .   .   .    ;  POSITIVE ADJUSTMENT - ADD INPUT
4685   103A   .   .   .    ;
4686   103A   .   .   .    CHK150 EQU $
4687   103A   86  .   .           ADD M        ;OVERFLOW?
4688   103B   D2  34  10          JNC CHK060   ;NO - USE SPECIFIED VALUE
4689   103E   C3  37  10          JMP CHK070   ;YES - USE MAXIMUM VALUE
4690   1041   .   .   .    ;
4691   1041   .   .   .    ;  NEGATIVE ADJUSTMENT - SUBTRACT INPUT
4692   1041   .   .   .    ;
4693   1041   .   .   .    CHK160 EQU $
4694   1041   96  .   .           SUB M        ;UNDERFLOW?
4695   1042   DA  25  10          JC   CHK010    ;YES - USE ZERO
4696   1045   12  .   .           STAX D          ;NO - USE COMPUTED VALUE
4697   1046   C9  .   .           RET             ;RETURN
```

```
4699   1047    .    .    .    ;*********************************************
4700   1047    .    .    .    ; CKDSPF - CHECK FOR DISPLAY FUNCTIONS ENABLED *
4701   1047    .    .    .    ;*********************************************
4702   1047    .    .    .    CKDSPF EQU   $
4703   1047    3A   F4   FF          LDA   MDFLG1      ;GET SOFT MODE FLAGS
4704   104A    E6   01   .           ANI   DSPFNC      ;MASK FOR DISPLAY FUNCTIONS
4705   104C    C9   .    .           RET                   ;FLAG AND RETURN
4706   104D    .    .    .    ;*************************************
4707   104D    .    .    .    ; CKEDIT - CHECK FOR EDIT MODE ENABLED *
4708   104D    .    .    .    ;*************************************
4709   104D    .    .    .    CKEDIT EQU   $
4710   104D    3A   F4   FF          LDA   MDFLG1      ;GET SOFT MODE FLAGS
4711   1050    E6   10   .           ANI   EDIT        ;MASK FOR EDIT FLAG AND
4712   1052    C9   .    .           RET                   ;RETURN
4713   1053    .    .    .    ;*************************************
4714   1053    .    .    .    ; GTMODE - DETERMINE MODE OF TERMINAL *
4715   1053    .    .    .    ; Z = TRUE IF CHARACTER MODE          *
4716   1053    .    .    .    ; Z = FALSE IF PAGE MODE              *
4717   1053    .    .    .    ;*************************************
4718   1053    .    .    .    GTMOD1 EQU   $
4719   1053    3A   64   FF          LDA   IOFLG2      ;GET I/O FLAGS
4720   1056    E6   20   .           ANI   XDS2BF      ;DISPLAY TO BUFFER TRANSFER?
4721   1058    C0   .    .           RNZ               ;YES - RETURN PAGE MODE
4722   1059    .    .    .    GTMODE EQU   $           ;NO - CHECK REAL PAGE MODE
4723   1059    3A   F3   FF          LDA   MDFLG2      ;GET TERMINAL MODE FLAGS 2
4724   105C    E6   02   .           ANI   BLKMDE      ;BLOCK MODE ENABLED?
4725   105E    C8   .    .           RZ             ;NO - RETURN (Z=TRUE)
4726   105F    .    .    .    ;
4727   105F    .    .    .    ;  CKLNMD - CHECK LINE MODE
4728   105F    .    .    .    ;
4729   105F    .    .    .    ;      EXIT :  Z = TRUE, LINE MODE
4730   105F    .    .    .    ;                = FALSE, PAGE MODE
4731   105F    .    .    .    ;               A,L DESTROYED
4732   105F    .    .    .    ;
4733   105F    .    .    .    CKLNMD EQU $
4734   105F    3A   FB   FF          LDA   KBJMPR      ;GET THE STRAP SETTINGS
4735   1062    E6   08   .           ANI   PAGSTR      ;SET Z-FLAG
4736   1064    C9   .    .           RET               ;RETURN
```

```
 4738    1065   .   .   .   ;******************************************
 4739    1065   .   .   .   ; CKPROT - CHECK PROTECT STATUS OF CURRENT *
 4740    1065   .   .   .   ;    CURSOR LOCATION                       *
 4741    1065   .   .   .   ;******************************************
 4742    1065   .   .   .   CKPROT EQU  $
 4743    1065   3A  C2  FF         LDA  PROFLD     ;GET PROTECT FLAG
 4744    1068   3C  .   .          INR  A          ;SET Z-FLAG (-1 => PROTECTED
 4745    1069   C9  .   .          RET             ;RETURN
 4746    106A   .   .   .   ;****************************************
 4747    106A   .   .   .   ; CKRMTE - CHECK FOR REMOTE MODE ENABLED *
 4748    106A   .   .   .   ;****************************************
 4749    106A   .   .   .   CKRMTE EQU  $
 4750    106A   3A  F8  FF         LDA  CMFLGS     ;GET COMMON FLAGS
 4751    106D   E6  10  .          ANI  REMSET     ;MASK FOR REMOTE FLAG
 4752    106F   C9  .   .          RET             ;RET (NZ => YES; Z => NO)
```

```
=============================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                       PAGE 138
=============================================================================
 4754    1070    .   .   .   ;*******************************************
 4755    1070    .   .   .   ; CLBLXF - CLEAR BLOCK TRANSFER PENDING FLAG *
 4756    1070    .   .   .   ;*******************************************
 4757    1070    .   .   .   ;
 4758    1070    .   .   .   ;  ENTRY:  B = 377B-(FLAG TO CLEAR FROM MFLGS)
 4759    1070    .   .   .   ;          C = 377B-(FLAG TO CLEAR FROM MFLGS2)
 4760    1070    .   .   .   ;
 4761    1070    .   .   .   ;  EXIT :  H = BASEH
 4762    1070    .   .   .   ;          A,B,L DESTROYED
 4763    1070    .   .   .   ;
 4764    1070    .   .   .   ;  CLEARS THE SPECIFIED TRANSFER PENDING FLAG
 4765    1070    .   .   .   ;  FROM "MFLGS".  IF NO OTHER TRANSFER IS PENDING,
 4766    1070    .   .   .   ;  THEN THE KEYBOARD IS UNLOCKED.  OTHERWISE,
 4767    1070    .   .   .   ;  THE NEXT TRANSFER PENDING IS SET UP.
 4768    1070    .   .   .   ;
 4769    1070    .   .   .   CLBLXF EQU  $
 4770    1070    2A  6F  FF         LHLD MFLGS2     ;GET TRANSFER PENDING FLAGS
 4771    1073    78  .   .          MOV  A,B
 4772    1074    A4  .   .          ANA  H          ;CLEAR FLAG FROM "MFLGS"
 4773    1075    67  .   .          MOV  H,A
 4774    1076    79  .   .          MOV  A,C
 4775    1077    A5  .   .          ANA  L          ;CLEAR FLAG FROM "MFLGS2"
 4776    1078    6F  .   .          MOV  L,A
 4777    1079    22  6F  FF         SHLD MFLGS2     ;STORE NEW FLAG VALUES
 4778    107C    E6  03  .          ANI  SBINRY+SDVREC
 4779    107E    B4  .   .          ORA  H          ;ANY MORE TRANSFER PENDING?
 4780    107F    01  00  00         LXI  B,0          ;(SET FOR NULL FLAGS SET)
 4781    1082    C2  CA  16         JNZ  SBLXF0     ;YES - SET UP NEXT BLOCK XFR
 4782    1085    CD  F4  15         CALL KBEN       ;NO - RE-ENABLE KEYBOARD
 4783    1088    .   .   .   ;
 4784    1088    .   .   .   ;  CLRXON - CLEAR BLOCK TRANSFER TRIGGER
 4785    1088    .   .   .   ;
 4786    1088    .   .   .   CLRXON EQU  $
 4787    1088    3E  00  .          MVI  A,CLRTRG   ;CLEAR BLOCK TRANSFER TRIGGE
 4788    108A    CD  42  12         CALL DCMCTL     ;PERFORM DATACOM CONTROL
 4789    108D    37  .   .          STC             ;SET C-FLAG TRUE AND
 4790    108E    C9  .   .          RET               ;RETURN
```

```
4792    108F   .    .    .    ;********************************************
4793    108F   .    .    .    ; CLEARS - CLEAR DISPLAY FROM CURSOR POSITION *
4794    108F   .    .    .    ;********************************************
4795    108F   .    .    .    CLEARS EQU  $          ;CLEAR UNPROTECTED FIELDS
4796    108F   3E   FE   .           MVI  A,377Q-SDACOM  ;ONLY BY CLEARING DATA
4797    1091   CD   01   16          CALL CLRDFL       ;COMM INPUT FLAG
4798    1094   CD   76   19          CALL CHKFMS      ;FORMAT/SOFT KEY DEFINE MODE
4799    1097   C2   C9   10          JNZ  CLS100      ;YES - CLEAR FIELDS ONLY
4800    109A   CD   3C   1C          CALL CLEARL      ;CLEAR LINE FROM CURSOR
4801    109D   F8   .    .           RM               ;RETURN IF LINE NOT FOUND
4802    109E   2A   C9   FF          LHLD LSTLIN      ;GET CURRENT LINE ADDRESS
4803    10A1   7E   .    .           MOV  A,M         ;GET THE LSB VALUE
4804    10A2   B7   .    .           ORA  A           ;NEXT LINE EXIST (LSB # 0)?
4805    10A3   C8   .    .           RZ               ;NO - RETURN
4806    10A4   E5   .    .           PUSH H           ;YES - ADD SUCCEEDING LINES
4807    10A5   36   00   .           MVI  M,0           ;TO FREE BLOCKS LIST
4808    10A7   23   .    .           INX  H           ;SET NEXT LINE POINTER TO
4809    10A8   56   .    .           MOV  D,M           ;INDICATE NO NEXT LINE
4810    10A9   36   CE   .           MVI  M,EOP
4811    10AB   5F   .    .           MOV  E,A         ;SET D,E TO TOP NEXT LINE
4812    10AC   2A   AC   FF          LHLD FRBLKS      ;GET CURRENT FREE BLOCKS HEA
4813    10AF   EB   .    .           XCHG             ;SET PREVIOUS LINE POINTER
4814    10B0   23   .    .           INX  H             ;IN FIRST SUCCEEDING LINE
4815    10B1   23   .    .           INX  H             ;TO CURRENT FREE BLOCKS
4816    10B2   23   .    .           INX  H             ;HEAD
4817    10B3   73   .    .           MOV  M,E
4818    10B4   23   .    .           INX  H
4819    10B5   72   .    .           MOV  M,D
4820    10B6   2A   A1   FF          LHLD LLINE       ;SET FREE BLOCKS HEAD TO
4821    10B9   22   AC   FF          SHLD FRBLKS        ;CURRENT LAST LINE
4822    10BC   E1   .    .           POP  H           ;SET LAST LINE ADDRESS TO
4823    10BD   22   A1   FF          SHLD LLINE         ;CURRENT LINE
4824    10C0   .    .    .    ;*******************
4825    10C0   .    .    .    ; MEMORY RELEASED  *
4826    10C0   .    .    .    ; CLEAR LOCK FLAGS *
4827    10C0   .    .    .    ;*******************
4828    10C0   .    .    .    MLKOF   EQU $
4829    10C0   3A   F4   FF          LDA  MDFLG1
4830    10C3   E6   04   .           ANI  MEMLOK      ;MEMORY LOCK ENABLED?
4831    10C5   C8   .    .           RZ               ;NO - RETURN
4832    10C6   C3   D9   0A          JMP  ML0010      ;YES - SET LED ON WO/BLINKIN
```

```
 4834    10C9    .    .    .    ;****************************************
 4835    10C9    .    .    .    ; FORMAT MODE CLEAR SCREEN FROM CURSOR *
 4836    10C9    .    .    .    ;****************************************
 4837    10C9    .    .    .    CLS100 EQU  $
 4838    10C9    F4   CD   06          CP   RCADR4     ;LOCATE CHAR IF FORMAT MODE
 4839    10CC    F8   .    .           RM              ;RETURN IF NOT FOUND OR IN
 4840    10CD    .    .    .    ;                        SOFT KEY DEFINE MODE
 4841    10CD    C2   D4   10          JNZ  CLS110     ;PAST EOL - START AT NEXT FL
 4842    10D0    04   .    .           INR  B          ;CURSOR IN UNPROTECTED FIELD
 4843    10D1    C2   DD   10          JNZ  CLS130     ;YES - CLEAR REST OF FIELD
 4844    10D4    .    .    .    ;*********************************
 4845    10D4    .    .    .    ; CURSOR IN PROTECTED FIELD     *
 4846    10D4    .    .    .    ; TAB TO NEXT UNPROTECTED FIELD *
 4847    10D4    .    .    .    ;*********************************
 4848    10D4    .    .    .    CLS110 EQU $
 4849    10D4    CD   C4   1D          CALL FLDSR    ;SEARCH TO NEXT FIELD
 4850    10D7    C8   .    .           RZ            ;NO MORE FIELDS - RETURN
 4851    10D8    1A   .    .           LDAX D        ;GET END PROTECT CHARACTER
 4852    10D9    .    .    .    CLS120 EQU  $
 4853    10D9    32   C5   FF          STA  LSTFMT     ;SET LAST FORMAT CODE
 4854    10DC    1B   .    .           DCX  D          ;SKIP OVER "ENDPR" CHAR
 4855    10DD    .    .    .    ;***************************
 4856    10DD    .    .    .    ; CLEAR UNPROTECTED FIELD *
 4857    10DD    .    .    .    ;***************************
 4858    10DD    .    .    .    CLS130 EQU $
 4859    10DD    CD   9C   1C          CALL CLER01     ;CLEAR FIELD
 4860    10E0    FE   CE   .           CPI EOP        ;TERMINATION AT END OF PAGE?
 4861    10E2    C8   .    .           RZ             ;YES - RETURN
 4862    10E3    .    .    .    ;*************************************
 4863    10E3    .    .    .    ; SEARCH FOR NEXT UNPROTECTED FIELD *
 4864    10E3    .    .    .    ;*************************************
 4865    10E3    1B   .    .           DCX  D          ;ADJUST ADDRESS TO NEXT CHAR
 4866    10E4    .    .    .    CLS200 EQU  $
 4867    10E4    13   .    .           INX  D          ;ADJUST ADDRESS TO PREV CHAR
 4868    10E5    .    .    .    CLS210 EQU  $
 4869    10E5    CD   87   0B          CALL NXTCHR     ;GET NEXT CHARACTER
 4870    10E8    C2   E4   10          JNZ  CLS200     ;SKIP OVER EOL LINK
 4871    10EB    FE   C1   .           CPI  ENDPR      ;NEW FIELD?
 4872    10ED    CA   D9   10          JZ   CLS120     ;YES - CLEAR IT
 4873    10F0    FE   CE   .           CPI  EOP        ;END OF DISPLAY?
 4874    10F2    C8   .    .           RZ              ;YES - RETURN
 4875    10F3    FE   C4   .           CPI  STPFLG     ;NON-DISPLAYING TERMINATOR?
 4876    10F5    CC   B8   1A          CZ   CHRDL2     ;YES - DELETE IT
 4877    10F8    C3   E5   10          JMP  CLS210     ;CONTINUE SEARCH
```

```
===========================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS
===========================================================================
 4879    10FB    .    .    .    ;**************************
 4880    10FB    .    .    .    ; CLRALL - CLEAR ALL TABS *
 4881    10FB    .    .    .    ;**************************
 4882    10FB    .    .    .    ;
 4883    10FB    .    .    .    ;  ENTRY:  H = BASEH
 4884    10FB    .    .    .    ;
 4885    10FB    .    .    .    CLRALL EQU  $
 4886    10FB    2E   78   .          MVI  L,HTBTBL-BASE  ;SET ADDRESS AND NUMBER
 4887    10FD    1E   0A   .          MVI  E,HTBLEN     ;OF BYTES TO BE CLEARED
 4888    10FF    .    .    .    ;****************************************
 4889    10FF    .    .    .    ; CLRAL1 - SET A REGION OF RAM TO ZERO *
 4890    10FF    .    .    .    ;****************************************
 4891    10FF    .    .    .    ;
 4892    10FF    .    .    .    ;  ENTRY:  E = NUMBER OF BYTES IN REGION
 4893    10FF    .    .    .    ;          H,L = LOW ADDRESS OF REGION
 4894    10FF    .    .    .    ;
 4895    10FF    .    .    .    ;  EXIT :  A,E = 0
 4896    10FF    .    .    .    ;          H,L = H,L(ENTRY) + E
 4897    10FF    .    .    .    ;
 4898    10FF    .    .    .    CLRAL1 EQU  $
 4899    10FF    AF   .    .          XRA  A            ;SET A TO ZERO
 4900    1100    .    .    .    CLA010 EQU  $
 4901    1100    77   .    .          MOV  M,A          ;SET BYTE TO ZERO
 4902    1101    23   .    .          INX  H            ;ADVANCE TO NEXT BYTE
 4903    1102    1D   .    .          DCR  E            ;ALL BYTES DONE?
 4904    1103    C2   00   11         JNZ  CLA010       ;NO - DO NEXT BYTE
 4905    1106    C9   .    .          RET               ;YES - RETURN
```

```
4907   1107    .   .   .   ;*****************************************
4908   1107    .   .   .   ; CURPHD - HOME DOWN TO FIRST COLUMN OF *
4909   1107    .   .   .   ; FIRST LINE BEYOND END OF MEMORY       *
4910   1107    .   .   .   ;*****************************************
4911   1107    .   .   .   CURPHD EQU  $
4912   1107    CD  8C  19         CALL CHKSFK     ;DEFINE SOFT KEY MODE?
4913   110A    C0  .   .          RNZ             ;YES - IGNORE HOME DOWN
4914   110B    CD  C5  21         CALL CURPRT     ;NO - RETURN TO LEFT MARGIN
4915   110E    .   .   .   ;**************************
4916   110E    .   .   .   ; MOVE CURSOR TO NEXT ROW *
4917   110E    .   .   .   ;**************************
4918   110E    .   .   .   HDC100 EQU  $
4919   110E    3A  C7  FF         LDA  LSTROW
4920   1111    FE  17  .          CPI  MAXROW     ;IS LAST ROW DONE AT BOTTOM?
4921   1113    CC  27  0C         CZ   ROLLUP     ;YES - ROLL UP THE DISPLAY
4922   1116    2A  C9  FF         LHLD LSTLIN     ;GET CURRENT ROW ADDRESS
4923   1119    7E  .   .          MOV  A,M        ;GET LSB OF NEXT LINE POINTE
4924   111A    B7  .   .          ORA  A          ;IS THERE A NEXT ROW?
4925   111B    CA  2A  11         JZ   HDC200     ;NO - TERMINATE HOME DOWN
4926   111E    5F  .   .          MOV  E,A        ;YES - SET E TO NEXT LINE
4927   111F    1C  .   .          INR  E            ;POINTER OF NEXT LINE
4928   1120    CD  54  0C         CALL ROLUP2     ;SET "LSTLIN" AND "CURADR"
4929   1123    21  C7  FF         LXI  H,LSTROW     ;TO NEXT LINE
4930   1126    34  .   .          INR  M          ;INCREMENT LAST ROW DONE
4931   1127    C3  0E  11         JMP HDC100
4932   112A    .   .   .   ;******************
4933   112A    .   .   .   ; LAST LINE FOUND *
4934   112A    .   .   .   ; SET ROW
4935   112A    .   .   .   ;******************
4936   112A    .   .   .   HDC200 EQU  $
4937   112A    CD  86  0B         CALL NXTCHR     ;GET 1ST CHAR OF LAST ROW
4938   112D    FE  CC  .          CPI  EOL        ;LAST ROW EMPTY?
4939   112F    3A  C7  FF         LDA  LSTROW       ;(GET LAST ROW POSITION)
4940   1132    CA  36  11         JZ   HDC210     ;YES - SET CURRENT ROW = LAS
4941   1135    3C  .   .          INR  A          ;NO - SET TO NEXT ROW
4942   1136    .   .   .   HDC210 EQU  $
4943   1136    32  C0  FF         STA  CURROW     ;SET CURRENT ROW NUMBER
4944   1139    C9  .   .          RET             ;RETURN
```

```
=========================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS
=========================================================================
4946    113A    .    .    .    ;********************************
4947    113A    .    .    .    ; CURPOS - CURSOR POSITIONING  *
4948    113A    .    .    .    ;          INITIAL ENTRY POINT *
4949    113A    .    .    .    ;********************************
4950    113A    .    .    .    CURPOS EQU $
4951    113A    3A   C1   FF          LDA  CURCOL      ;SET NEW COLUMN DEFAULT TO
4952    113D    32   DB   FF          STA  NEWCOL        ;CURRENT COLUMN POSITION
4953    1140    3E   7F   .           MVI  A,377Q-NWRWST
4954    1142    CD   AA   04          CALL CLRMF2      ;CLEAR NEW ROW SET FLAG
4955    1145    2E   D9   .           MVI  L,SCRNRW-BASE  ;PRESET RELATIVE ROW
4956    1147    36   FF   .           MVI  M,-1          ;PARAMETER TO -1
4957    1149    21   60   27          LXI  H,CRPTAB   ;SET RANGE TABLE FOR CURSOR
4958    114C    C3   7F   04          JMP  ESCAPA        ;POSITIONING
```

```
 4960   114F    .    .    .   ;**********************************
 4961   114F    .    .    .   ; NEW COLUMN POSITION IS DEFINED *
 4962   114F    .    .    .   ;**********************************
 4963   114F    .    .    .   CURPO1 EQU  $
 4964   114F    0E   4F   .          MVI  C,MAXCOL  ;SET MAXIMUM VALUE AND
 4965   1151    11   DB   FF         LXI  D,NEWCOL     ;PARAMETER TO BE SET
 4966   1154    2E   C1   .          MVI  L,CURCOL-BASE    ;"CHKLIM"
 4967   1156    46   .    .          MOV B,M
 4968   1157    C3   7D   11         JMP CRP025    ;EVALUATE THE PARAMETER
 4969   115A    .    .    .   ;***********************
 4970   115A    .    .    .   ; SCREEN ROW SPECIFIED *
 4971   115A    .    .    .   ;***********************
 4972   115A    .    .    .   CURPO2 EQU  $
 4973   115A    0E   17   .          MVI  C,MAXROW  ;SET MAXIMUM VALUE AND
 4974   115C    11   D9   FF         LXI  D,SCRNRW     ;PARAMETER TO BE SET
 4975   115F    2E   C0   .          MVI  L,CURROW-BASE     ;"CHKLIM"
 4976   1161    46   .    .          MOV B,M
 4977   1162    C3   7D   11         JMP CRP025     ;EVALUATE THE PARAMETER
 4978   1165    .    .    .   ;*******************************
 4979   1165    .    .    .   ; NEW ROW POSITION IS DEFINED *
 4980   1165    .    .    .   ;*******************************
 4981   1165    .    .    .   CURPO3 EQU  $
 4982   1165    3A   6B   FF         LDA  MLKROW       ;GET MEMORY LOCK ROW
 4983   1168    B7   .    .          ORA  A            ;MEMORY LOCK ENABLED?
 4984   1169    C2   80   11         JNZ  CRP050       ;YES - IGNORE PARAMETER
 4985   116C    3E   80   .          MVI  A,NWRWST  ;NO - SET NEW ROW SET
 4986   116E    CD   39   17         CALL SETMF2       ;FLAG
 4987   1171    0E   FF   .          MVI  C,255     ;SET MAXIMUM VALUE AND
 4988   1173    11   DA   FF         LXI  D,NEWROW     ;PARAMETER TO BE SET
 4989   1176    3A   C0   FF         LDA  CURROW    ;COMPUTE CURRENT ABSOLUTE
 4990   1179    2E   A3   .          MVI  L,TLINO      ;ROW ADDRESS
 4991   117B    86   .    .          ADD  M
 4992   117C    47   .    .          MOV  B,A          ;PUT IT INTO B-REGISTER
 4993   117D    .    .    .   CRP025 EQU  $
 4994   117D    CD   11   10         CALL CHKL10       ;EVALUATE INPUT PARAMETER
 4995   1180    .    .    .   CRP050 EQU  $
 4996   1180    3A   88   FF         LDA  CHAR         ;RECALL THE INPUT CHARACTER
 4997   1183    E6   20   .          ANI  40Q          ;IS IT AN UPPER CASE CHAR?
 4998   1185    C2   87   04         JNZ  ESCAPB       ;NO - CONTINUE ESC SEQUENCE
 4999   1188    .    .    .   ;                       YES - POSITION CURSOR
```

```
==========================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                      PAGE 145
==========================================================================
5001    1188    .    .    .   ;******************************
5002    1188    .    .    .   ; EXECUTE COMPLETED SEQUENCE *
5003    1188    .    .    .   ;******************************
5004    1188    3A   D9   FF          LDA  SCRNRW  ;GET SCREEN ROW PARAMETER
5005    118B    B7   .    .           ORA A        ;WAS SCREEN ROW ADDRESS SET?
5006    118C    FA   92   11          JM  CRP200   ;NO - SET ABSOLUTE ROW ADDR
5007    118F    32   C0   FF          STA  CURROW  ;YES - SET NEW SCREEN ROW
5008    1192    .    .    .   ;***************************
5009    1192    .    .    .   ; SET ABSOLUTE ROW ADDRESS *
5010    1192    .    .    .   ;***************************
5011    1192    .    .    .   CRP200 EQU  $
5012    1192    FC   A6   11          CM   CRP500    ;FIND LOCATION OF NEW ROW
5013    1195    .    .    .   ;
5014    1195    .    .    .   ;  SET COLUMN ADDRESS
5015    1195    .    .    .   ;
5016    1195    3A   DB   FF          LDA  NEWCOL  ;GET NEW COLUMN ADDRESS
5017    1198    .    .    .   ;*******************************
5018    1198    .    .    .   ; LOCATE ADDRESS OF CHARACTER *
5019    1198    .    .    .   ;*******************************
5020    1198    .    .    .   CURP04 EQU  $
5021    1198    32   C1   FF          STA  CURCOL  ;STORE NEW COLUMN ADDRESS
5022    119B    CD   08   07          CALL RCADDR  ;FIND CHARACTER
5023    119E    C8   .    .           RZ           ;CHARACTER FOUND - RETURN
5024    119F    .    .    .   ;**********************************
5025    119F    .    .    .   ; CHARACTER NOT CURRENTLY STORED  *
5026    119F    .    .    .   ; BUILD LINE OVER TO NEW POSITION *
5027    119F    .    .    .   ;**********************************
5028    119F    2E   89   .           MVI L,DCHAR  ;SET UP BLANK FOR NEW POS.
5029    11A1    36   20   .           MVI M,ABLNK
5030    11A3    C3   A5   22          JMP DISPLO   ;BUILD BLOCKS
```

```
 5032   11A6    .    .    .   ;***********************************
 5033   11A6    .    .    .   ; LOCATE NEW ABSOLUTE ROW LOCATION *
 5034   11A6    .    .    .   ;***********************************
 5035   11A6    .    .    .   CRP500 EQU  $
 5036   11A6    3A   6F   FF         LDA  MFLGS2    ;GET TERMINAL MODE FLAGS
 5037   11A9    E6   80   .          ANI  NWRWST    ;NEW ABSOLUTE ROW SET?
 5038   11AB    C8   .    .          RZ             ;NO - RETURN
 5039   11AC    3A   DA   FF         LDA  NEWROW   ;GET NEW ROW VALUE
 5040   11AF    2E   A3   .          MVI L,TLINO  ;SUBTRACT ROW CORRESP.
 5041   11B1    96   .    .          SUB M        ;TO TOP OF PAGE
 5042   11B2    2E   C0   .          MVI L,CURROW
 5043   11B4    DA   B4   0B         JC  PRVPG1   ;LOCATE PREVIOUS ROW PAGE
 5044   11B7    FE   18   .          CPI MAXROW+1
 5045   11B9    77   .    .          MOV M,A      ;SET NEW ROW
 5046   11BA    D8   .    .          RC           ;RETURN IF SAME PAGE
 5047   11BB    .    .    .   ;******************************
 5048   11BB    .    .    .   ; ROW IS AFTER BOTTOM OF PAGE *
 5049   11BB    .    .    .   ; ROLL DISPLAY UP             *
 5050   11BB    .    .    .   ;******************************
 5051   11BB    36   17   .          MVI M,MAXROW ;SET ROW
 5052   11BD    D6   17   .          SUI  MAXROW    ;SET ROLL COUNT
 5053   11BF    .    .    .   STR010 EQU  $
 5054   11BF    CD   45   0B         CALL NXTPG1    ;ROLL DISPLAY UP
 5055   11C2    7E   .    .          MOV  A,M       ;GET NUMBER OF ROWS TO ROLL
 5056   11C3    91   .    .          SUB C        ;SUBTRACT ROWS ROLLED
 5057   11C4    C8   .    .          RZ           ;RETURN IF ROLL COMPLETE
 5058   11C5    77   .    .          MOV  M,A       ;SAVE NUMBER OF ROW TO ROLL
 5059   11C6    AF   .    .          XRA  A            ;(SET TO FIND COLUMN 0)
 5060   11C7    CD   0B   07         CALL RCADR0    ;BUILD NEW ROWS
 5061   11CA    C0   .    .          RNZ          ;RETURN IF OUT OF MEMORY
 5062   11CB    3A   83   FF         LDA  NMROLL    ;GET # OF ROWS TO ROLL
 5063   11CE    C3   BF   11         JMP  STR010    ;ROLL AGAIN
 5064   11D1    .    .    .   ;*********************************
 5065   11D1    .    .    .   ; CURSEN - CURSOR POSITION SENSE *
 5066   11D1    .    .    .   ;*********************************
 5067   11D1    .    .    .   ;
 5068   11D1    .    .    .   ;  RLCRSN - SCREEN RELATIVE CURSOR SENSE
 5069   11D1    .    .    .   ;
 5070   11D1    .    .    .   RLCRSN EQU $
 5071   11D1    3E   04   .          MVI  A,RELSNS  ;SET RELATIVE SENSE FLAG
 5072   11D3    CD   39   17         CALL SETMF2
 5073   11D6    C3   DE   11         JMP  CUR100    ;GO SET CURSOR SENSE FLAG
 5074   11D9    .    .    .   ;
 5075   11D9    .    .    .   ;  CURSEN - ABSOLUTE CURSOR SENSE
 5076   11D9    .    .    .   ;
 5077   11D9    .    .    .   CURSEN EQU  $
 5078   11D9    3E   FB   .          MVI  A,377Q-RELSNS
 5079   11DB    CD   AA   04         CALL CLRMF2    ;CLEAR RELATIVE SENSE FLAG
 5080   11DE    .    .    .   CUR100 EQU  $
 5081   11DE    01   00   10         LXI  B,SCRSEN  ;SET UP BLOCK TRANSFER
 5082   11E1    C3   CA   16         JMP  SBLXF0      ;FOR CURSOR SENSE PENDING
```

```
==============================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                          PAGE 147
==============================================================================
5084   11E4    .   .   .    CRSNGO EQU  $
5085   11E4    01  FF  EF          LXI  B,-1-SCRSEN  ;CLEAR CURSOR SENSE
5086   11E7    CD  70  10          CALL CLBLXF        ;PENDING FLAG
5087   11EA    06  26  .           MVI  B,AMPSND  ;SEND <ESC>-<&>
5088   11EC    CD  BB  17          CALL ESCOUT
5089   11EF    3E  61  .           MVI  A,SMALLA  ;TRANSMIT LOWER CASE A
5090   11F1    CD  C1  17          CALL XPUTDC
5091   11F4    .   .   .    ;************************
5092   11F4    .   .   .    ; OUTPUT CURSOR COLUMN *
5093   11F4    .   .   .    ;************************
5094   11F4    21  C1  17          LXI  H,XPUTDC  ;SEND NUMBER TO DATA COMM
5095   11F7    3A  C1  FF          LDA  CURCOL    ;GET CURRENT CURSOR COLUMN
5096   11FA    CD  23  08          CALL BN2DE1    ;CONVERT AND TRANSMIT VALUE
5097   11FD    3E  63  .           MVI  A,ALCC    ;TRANSMIT LOWER CASE C
5098   11FF    CD  C1  17          CALL XPUTDC
5099   1202    .   .   .    ;*********************
5100   1202    .   .   .    ; OUTPUT CURSOR ROW *
5101   1202    .   .   .    ;*********************
5102   1202    3A  6F  FF          LDA  MFLGS2    ;GET TERMINAL MODE FLAGS
5103   1205    E6  04  .           ANI  RELSNS    ;SCREEN RELATIVE SENSING?
5104   1207    3A  C0  FF          LDA  CURROW      ;(GET CURSOR ROW NUMBER)
5105   120A    06  59  .           MVI  B,Y         ;(SET DEFAULT PARAMETER)
5106   120C    C2  15  12          JNZ  CRS100    ;YES - OUTPUT SCREEN ADDRESS
5107   120F    21  A3  FF          LXI  H,TLINO   ;NO - COMPUTE ABSOLUTE
5108   1212    86  .   .           ADD  M           ;ROW NUMBER
5109   1213    06  52  .           MVI  B,R       ;SET ABSOLUTE PARAMETER CHAR
5110   1215    .   .   .    ;**************************
5111   1215    .   .   .    ; TRANSMIT ROW PARAMETER *
5112   1215    .   .   .    ;**************************
5113   1215    .   .   .    ;
5114   1215    .   .   .    ;  A = ROW VALUE
5115   1215    .   .   .    ;  B = ROW PARAMETER LETTER
5116   1215    .   .   .    ;
5117   1215    .   .   .    CRS100 EQU  $
5118   1215    C5  .   .           PUSH B         ;SAVE ROW PARAMETER LETTER
5119   1216    CD  26  08          CALL BN2DE2    ;CONVERT AND TRANSMIT VALUE
5120   1219    F1  .   .           POP  PSW       ;RECALL ROW PARAMETER LETTER
5121   121A    CD  C1  17          CALL XPUTDC    ;TRANSMIT ROW PARAMETER CHAR
5122   121D    .   .   .    ;                     FALL INTO "SDTERM"
5123   121D    .   .   .    ;************************************
5124   121D    .   .   .    ; SDTERM - SEND BLOCK TERMINATOR *
5125   121D    .   .   .    ; RS IF PAGE MODE, OTHERWISE CR(LF) *
5126   121D    .   .   .    ;************************************
5127   121D    .   .   .    SDTERM EQU  $
5128   121D    CD  F6  16          CALL SDTRM1    ;SEND TERMINATOR
5129   1220    .   .   .    SDTER1 EQU  $
5130   1220    CD  88  10          CALL CLRXON    ;CLEAR BLOCK TERMINATOR
5131   1223    3E  07  .           MVI  A,ENDBLK  ;TELL DATA COMM THAT LAST
5132   1225    C3  42  12          JMP  DCMCTL      ;CHARACTER IN BLOCK IS OUT
```

```
==========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
==========================================================================
 5134   1228    .    .    .   ;
 5135   1228    .    .    .   ;   DC2GO - OUTPUT DC2
 5136   1228    .    .    .   ;
 5137   1228    .    .    .   DC2GO   EQU  $
 5138   1228   21   70   FF           LXI  H,MFLGS
 5139   122B   7E    .    .           MOV  A,M         ;CLEAR DC2 PENDING FLAG
 5140   122C   E6   FE    .           ANI  (-1-SDC2)/256
 5141   122E   77    .    .           MOV  M,A
 5142   122F   3E   0D    .           MVI  A,PROMPT    ;TELL DATA COMM ROUTINE TO
 5143   1231   C3   42   12           JMP  DCMCTL        ;SEND PROMPT CODE
 5144   1234    .    .    .   ;****************************************
 5145   1234    .    .    .   ; DCMINT - DATA COMM INTERRUPT ROUTINE *
 5146   1234    .    .    .   ;****************************************
 5147   1234    .    .    .   ;
 5148   1234    .    .    .   ;   ENTRY:  PSW "PUSHED"
 5149   1234    .    .    .   ;           A = INTERRUPT CODE
 5150   1234    .    .    .   ;
 5151   1234    .    .    .   DCMINT EQU   $
 5152   1234   CD   65   91          CALL INTVEC      ;CHECK ALTERNATE INTERRUPT
 5153   1237   F1    .    .          POP  PSW         ;RESTORE PSW AND A-REGISTER
 5154   1238   C3   26   50          JMP  ZDCINT      ;EXECUTE NORMAL DATA COMM
 5155   123B    .    .    .   ;                          INTERRUPT ROUTINE
```

```
123B   .  .  .   ;**********************************
123B   .  .  .   ; BRKDC - EXECUTE DATA COMM BREAK *
123B   .  .  .   ;**********************************
123B   .  .  .   BRKDC  EQU  $
123B   3E 05 .          MVI  A,PUTBRK  ;EXECUTE DATACOM BREAK
123D   C3 42 12         JMP  DCMCTL       ;CONTROL
1240   .  .  .   ;
1240   .  .  .   ;  DISMDM - DISCONNECT MODEM
1240   .  .  .   ;
1240   .  .  .   DISMDM EQU  $
1240   3E 06 .          MVI  A,DISCNT  ;EXECUTE MODEM DISCONNECT
1242   .  .  .   ;**********************************************
1242   .  .  .   ; DCMCTL - PERFORM DATA COMM CONTROL FUNCTION *
1242   .  .  .   ;**********************************************
1242   .  .  .   ;
1242   .  .  .   ;  ENTRY:  A = FUNCTION TYPE NUMBER
1242   .  .  .   ;
1242   .  .  .   ;  EXIT :  Z - FUNCTION PERFORMED
1242   .  .  .   ;         NZ - FUNCTION NOT PERFORMED
1242   .  .  .   ;
1242   .  .  .   DCMCTL EQU  $
1242   F5 .  .          PUSH PSW       ;SAVE A-REGISTER
1243   CD 6A 10         CALL CKRMTE    ;REMOTE MODE ENABLED?
1246   C2 4C 12         JNZ  DCC010    ;YES - PERFORM FUNCTION
1249   F1 .  .          POP  PSW       ;NO - RESTORE A-REGISTER
124A   3C .  .          INR  A         ;FORCE NZ
124B   C9 .  .          RET            ;RETURN
124C   .  .  .   ;
124C   .  .  .   DCC010 EQU  $
124C   F1 .  .          POP  PSW       ;RESTORE A-REGISTER
124D   .  .  .   DCMC11 EQU  $         ;ENTRY TO FORCE DATA COMM CT
124D   CD 11 50         CALL ZDCCTL    ;EXECUTE FUNCTION
1250   D0 .  .          RNC            ;SUCCESSFUL - RETURN
1251   .  .  .   DCERR  EQU  $         ;PROCESS DATA COMM ERROR
1251   CA 14 48         JZ   ZBELL     ;NOT FATAL - SOUND BELL
1254   .  .  .   ;****************************
1254   .  .  .   ; DISPLAY TEST FAIL MESSAGES *
1254   .  .  .   ;****************************
1254   .  .  .   HANG00 EQU  $
1254   CD D6 1C         CALL DSPMS0    ;DISPLAY THE ERROR MESSAGE
1257   3E 04 .          MVI  A,FRCRST  ;SET TO FORCE FULL RESET
1259   CD 00 14         CALL STCMFL      ;IF RESET KEY HIT
125C   .  .  .   ;
125C   .  .  .   HNG010 EQU  $
125C   CD A5 0F         CALL DISLN4    ;RE-ENABLE RESET ONLY
125F   C3 5C 12         JMP  HNG010    ;HANG TERMINAL
1262   .  .  .   ;                     ************************
1262   .  .  .   ;                     * RESET KEY MUST BE HIT *
1262   .  .  .   ;                     * TO RESTORE TERMINAL   *
1262   .  .  .   ;                     * OPERATION             *
1262   .  .  .   ;                     ************************
```

```
 5209   1262    .   .   .    ;*******************************************
 5210   1262    .   .   .    ; DCNUM - ACCUMULATE PARAMETER FOR ESC SEQ *
 5211   1262    .   .   .    ;*******************************************
 5212   1262    .   .   .    ;
 5213   1262    .   .   .    ;  EXIT :  Z TRUE
 5214   1262    .   .   .    ;
 5215   1262    .   .   .    DCNUM  EQU  $
 5216   1262    21  DD  FF          LXI  H,IOCSGN  ;GET THE CURRENT SIGN
 5217   1265    7E  .   .           MOV  A,M         ;VALUE
 5218   1266    B7  .   .           ORA  A         ;HAS ANY SIGN BEEN SET?
 5219   1267    C2  6C  12          JNZ  DCN005    ;YES - DON'T CHANGE IT
 5220   126A    36  80  .           MVI  M,NOSIGN  ;NO - SET NO SIGN FLAG
 5221   126C    .   .   .    DCN005 EQU  $
 5222   126C    3A  88  FF          LDA  CHAR      ;GET INPUT CHARACTER
 5223   126F    D6  30  .           SUI  ZERO      ;EXTRACT BINARY VALUE
 5224   1271    5F  .   .           MOV  E,A       ;PUT VALUE IN E-REGISTER
 5225   1272    16  00  .           MVI  D,0       ;SET MSB TO ZERO
 5226   1274    3A  D4  FF          LDA  RADIX     ;GET RADIX OF NUMBER
 5227   1277    2A  DE  FF          LHLD IODATA    ;GET ACCUMULATOR
 5228   127A    EB  .   .           XCHG           ;PUT ACCUMULATOR IN D,E
 5229   127B    .   .   .    ;
 5230   127B    .   .   .    DCN010 EQU  $
 5231   127B    19  .   .           DAD  D         ;ACCUMULATE NEW VALUE
 5232   127C    3D  .   .           DCR  A         ;RADIX ADJUSTMENT COMPLETED?
 5233   127D    C2  7B  12          JNZ  DCN010    ;NO - CONTINUE ADDING
 5234   1280    22  DE  FF          SHLD IODATA    ;YES - STORE NEW VALUE
 5235   1283    C3  8F  04          JMP  ESCAP1    ;CONTINUE ESCAPE SEQUENCE
 5236   1286    .   .   .    ;********************************************
 5237   1286    .   .   .    ; DCPLUS - PLUS SIGN RECEIVED FOR PARAMETER *
 5238   1286    .   .   .    ;********************************************
 5239   1286    .   .   .    DCPLUS EQU  $
 5240   1286    06  01  .           MVI  B,1       ;SET B-REG TO SIGN VALUE
 5241   1288    C3  8D  12          JMP  DCM010    ;SET SIGN FLAG
 5242   128B    .   .   .    ;*********************************************
 5243   128B    .   .   .    ; DCMNUS - MINUS SIGN RECEIVED FOR PARAMETER *
 5244   128B    .   .   .    ;*********************************************
 5245   128B    .   .   .    DCMNUS EQU  $
 5246   128B    06  FF  .           MVI  B,-1
 5247   128D    .   .   .    DCM010 EQU  $
 5248   128D    21  DD  FF          LXI  H,IOCSGN  ;GET CURRENT SIGN VALUE
 5249   1290    7E  .   .           MOV  A,M
 5250   1291    B7  .   .           ORA  A         ;SIGN SET ALREADY?
 5251   1292    C2  95  04          JNZ  ESCEND    ;YES - ABORT ESCAPE SEQUENCE
 5252   1295    70  .   .           MOV  M,B       ;NO - SET SIGN VALUE
 5253   1296    C3  8F  04          JMP  ESCAP1    ;CONTINUE ESCAPE SEQUENCE
```

```
=========================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                     PAGE 151
=========================================================================
5255   1299   .   .   .    ;****************************************
5256   1299   .   .   .    ; DCTEST - EXECUTE DATA COMM SELF-TEST *
5257   1299   .   .   .    ;****************************************
5258   1299   .   .   .    DCTEST EQU  $
5259   1299   CD  6A  10          CALL CKRMTE     ;REMOTE MODE ENABLED?
5260   129C   C8  .   .           RZ              ;NO - DON'T DO SELF-TEST
5261   129D   CD  14  50          CALL ZDCTST     ;CALL DATA COMM SELF-TEST
5262   12A0   DA  54  12          JC   HANGU0     ;HANG TERMINAL IF FATAL ERRO
5263   12A3   C3  D7  1C          JMP  DSPMS1     ;DISPLAY MESSAGE AND EXIT
5264   12A6   .   .   .    ;                         IF SELF-TEST SUCCESSFUL
5265   12A6   .   .   .    ;*****************************************
5266   12A6   .   .   .    ; DCXB2D - SEE IF SOURCE OF CHARACTER IS *
5267   12A6   .   .   .    ;   DATA COMM OR I/O BUFFER              *
5268   12A6   .   .   .    ;*****************************************
5269   12A6   .   .   .    ;
5270   12A6   .   .   .    ;  EXIT :  Z - INPUT IS NOT FROM DATA COMM OR I/O
5271   12A6   .   .   .    ;          NZ - INPUT IS FROM DATA COMM OR I/O
5272   12A6   .   .   .    ;          A DESTROYED
5273   12A6   .   .   .    ;
5274   12A6   .   .   .    DCXB2D EQU  $
5275   12A6   3A  6E  FF          LDA  DFLGS      ;GET DATA TRANSFER FLAGS
5276   12A9   E6  81  .           ANI  SDACOM+XBF2DS  ;SET Z-FLAG
5277   12AB   C9  .   .           RET             ;RETURN
```

```
 5279   12AC   .   .   .   ;******************************
 5280   12AC   .   .   .   ; DELAY0 - PAUSE FOR 1 SECOND *
 5281   12AC   .   .   .   ;******************************
 5282   12AC   .   .   .   DELAY0 EQU  $
 5283   12AC   3E  18  .          MVI  A,MAXROW+1  ;REMOVE CURSOR AND
 5284   12AE   CD  A1  0F         CALL DISLN2      ;RE-ENABLE RESET KEY
 5285   12B1   2E  64  .          MVI  L,100     ;DELAY FOR 1 SECOND
 5286   12B3   .   .   .   ;********************************
 5287   12B3   .   .   .   ; DELAY - DELAY 10 MILLISECONDS *
 5288   12B3   .   .   .   ; TIMES COUNT IN L              *
 5289   12B3   .   .   .   ;********************************
 5290   12B3   .   .   .   DELAY  EQU  $
 5291   12B3   3E  80  .          MVI  A,SETROM
 5292   12B5   D3  70  .          OUT  PROCSR     ;RESET THE TIMER
 5293   12B7   3A  F5  FF         LDA  PRCCTL     ;RESTORE PROCESSOR STATE
 5294   12BA   D3  70  .          OUT  PROCSR
 5295   12BC   .   .   .   ;
 5296   12BC   .   .   .   DLY010 EQU  $
 5297   12BC   AF  .   .          XRA  A          ;CLEAR THE INTERRUPT FLAG
 5298   12BD   32  F6  FF         STA  INTFLG
 5299   12C0   .   .   .   DLY020 EQU  $
 5300   12C0   76  .   .          HLT             ;SLEEP UNTIL INTERRUPTED
 5301   12C1   3A  F6  FF         LDA  INTFLG     ;GET INTERRUPT FLAG
 5302   12C4   FE  03  .          CPI  TMRINT     ;TIMER INTERRUPT?
 5303   12C6   C2  C0  12         JNZ  DLY020     ;NO - CONTINUE WAITING
 5304   12C9   2D  .   .          DCR L          ;ENOUGH TIMER INTERRUPTS?
 5305   12CA   C2  BC  12         JNZ DLY010     ;NO - CONTINUE TIMING
 5306   12CD   C9  .   .          RET             ;YES - RETURN
```

```
5308   12CE    .   .   .    ;***************************
5309   12CE    .   .   .    ; <F> - SEND FUNCTION DATA *
5310   12CE    .   .   .    ;***************************
5311   12CE    .   .   .    SNDCD2 EQU  $
5312   12CE    3A  DE  FF          LDA  IODATA     ;GET ACCUMULATED VALUE
5313   12D1    47  .   .           MOV  B,A        ;PUT CODE INTO B-REGISTER
5314   12D2    3E  0C  .           MVI  A,SNDFCT   ;SET DATA COMM CONTROL CODE
5315   12D4    CD  42  12          CALL DCMCTL     ;PERFORM FUNCTION
5316   12D7    C0  .   .           RNZ             ;EXIT IF FUNCTION NOT DONE
5317   12D8    .   .   .    ;                       OTHERWISE, SEND SCREEN DATA
5318   12D8    .   .   .    ;***************
5319   12D8    .   .   .    ; DISPLAY SEND *
5320   12D8    .   .   .    ;***************
5321   12D8    .   .   .    DPSEND EQU  $
5322   12D8    3E  08  .           MVI  A,CKIOKY
5323   12DA    CD  08  48          CALL ZKBCTL     ;I/O CONTROL KEY DOWN ALSO?
5324   12DD    C2  99  12          JNZ  DCTEST     ;YES - DO DATA COMM SELF-TES
5325   12E0    3A  F8  FF          LDA  CMFLGS     ;GET COMMON FLAGS
5326   12E3    E6  10  .           ANI  REMSET     ;REMOTE ENABLED?
5327   12E5    11  0A  00          LXI  D,(RCKYCD-ENTRCD)*2;(SET KEY INDEX)
5328   12E8    CA  8C  15          JZ   IOKEYS     ;NO - PERFORM RECORD COMMAND
5329   12EB    01  00  40          LXI  B,SENTER   ;YES - SET XFR PENDING FLAG
5330   12EE    3A  F3  FF          LDA  MDFLG2     ;NO - GET TERMINAL MODE FLAG
5331   12F1    E6  02  .           ANI  BLKMDE     ;BLOCK MODE ENABLED?
5332   12F3    CA  0E  13          JZ   DPS200     ;NO - DO CHARACTER MODE STAR
5333   12F6    CD  8C  19          CALL CHKSFK     ;SOFT KEY MODE?
5334   12F9    C2  22  13          JNZ  DPS220     ;YES - DON'T SET TERMINATOR
5335   12FC    3A  FA  FF          LDA  KBJMP2     ;YES - GET KEYBOARD JUMPERS
5336   12FF    E6  01  .           ANI  AUTTRM     ;AUTO TERMINATE ENABLED?
5337   1301    CA  22  13          JZ   DPS220     ;NO - DO DON'T MOVE CURSOR
5338   1304    CD  71  17          CALL STTERM     ;YES - SET NON-DISPLAYING
5339   1307    .   .   .    ;                         TERMINATOR
5340   1307    C8  .   .           RZ              ;EXIT IF NOT SUCCESSFUL
5341   1308    .   .   .    ;*****************************************
5342   1308    .   .   .    ; FIRST TRANSMIT CHARACTER LOCATED - SET *
5343   1308    .   .   .    ;   TRANSFER PENDING FLAG                *
5344   1308    .   .   .    ;*****************************************
5345   1308    .   .   .    DPS100 EQU  $
5346   1308    01  00  40          LXI  B,SENTER   ;SET B,C XFR PENDING FLAG
5347   130B    C3  D5  16          JMP  SBLXF1        ;FOR BLOCK MODE TRANSFER
```

```
5349    130E   .   .   .    ;******************************************
5350    130E   .   .   .    ; AUTO TERMINATOR JUMPER NOT REMOVED - DO *
5351    130E   .   .   .    ;   NORMAL DATA ENTRY FROM DISPLAY         *
5352    130E   .   .   .    ;******************************************
5353    130E   .   .   .    DPS200 EQU  $
5354    130E   3A  FB  FF          LDA  KBJMPR     ;GET KEYBOARD JUMPERS 1
5355    1311   E6  C0  .           ANI  HNDSHK+DC2SND
5356    1313   EE  40  .           XRI  HNDSHK     ;HANDSHAKE ENABLED?
5357    1315   C4  24  04          CNZ  CHKCT1     ;NO - SET BLOCK TRIGGER
5358    1318   .   .   .    DPS210 EQU  $
5359    1318   CD  CD  16          CALL SBLXFA     ;SET TRANSFER PENDING FLAG
5360    131B   .   .   .    DPS215 EQU  $
5361    131B   CD  76  19          CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
5362    131E   C0  .   .           RNZ             ;YES - DON'T MOVE CURSOR
5363    131F   C3  C8  21          JMP  CRRET1     ;NO - PUT CURSOR AT BEGINNIN
5364    1322   .   .   .    ;                        OF LINE (A = 0)
5365    1322   .   .   .    ;
5366    1322   .   .   .    ;  SET KEYBOARD BLOCK TRANSFER
5367    1322   .   .   .    ;
5368    1322   .   .   .    DPS220 EQU  $
5369    1322   CD  D5  16          CALL SBLXF1     ;SET BLOCK MODE XFR PENDING
5370    1325   3A  70  FF          LDA  MFLGS      ;GET TRANSFER PENDING FLAGS
5371    1328   E6  01  .           ANI  SDC2/256   ;DC2 TO BE SENT?
5372    132A   C0  .   .           RNZ             ;YES - DON'T MOVE CURSOR
5373    132B   CD  5F  10          CALL CKLNMD     ;LINE MODE?
5374    132E   CA  1B  13          JZ   DPS215     ;YES - SET CURSOR IN LINE
5375    1331   .   .   .    ;****************************************
5376    1331   .   .   .    ; DPSEN1 - HOME CURSOR FOR TRANSMISSION *
5377    1331   .   .   .    ;****************************************
5378    1331   .   .   .    DPSEN1 EQU  $
5379    1331   CD  F4  17          CALL XMOHME     ;HOME CURSOR
5380    1334   CD  76  19          CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
5381    1337   C0  .   .           RNZ             ;YES - RETURN
5382    1338   C3  C8  21          JMP  CRRET1     ;NO - SET CURSOR TO COLUMN 0
```

```
5384    133B     .   .   .   ;
5385    133B     .   .   .   ; * * * * * * * * * * * * * * * * * * * * * * * * *
5386    133B     .   .   .   ;
5387    133B     .   .   .   ;  DPSGO - SEND DISPLAY TO DATACOM
5388    133B     .   .   .   ;
5389    133B     .   .   .   ;      ENTRY:  CURCOL,CURROW SET TO STARTING
5390    133B     .   .   .   ;                   LOCATION
5391    133B     .   .   .   ;
5392    133B     .   .   .   ;      EXIT :  ALL REGISTERS DESTROYED
5393    133B     .   .   .   ;
5394    133B     .   .   .   DPSGO  EQU  $
5395    133B    CD  9C  25          CALL INITDG    ;INIT DISPLAY GET ROUTINE
5396    133E    C2  8A  13          JNZ  DSG200    ;TERMINATE IF NO CHARACTERS
5397    1341    3E  FF   .          MVI  A,STPXFR  ;SET TERMINATOR FUNCTION TO
5398    1343    32  6D  FF          STA  TRMFCT      ;TERMINATE TRANSFER
5399    1346     .   .   .   DSG010 EQU  $
5400    1346    3E  0D   .          MVI  A,STCHST  ;SET CHARACTER SET FOR
5401    1348    CD  08  48          CALL ZKBCTL      ;FOREIGN TERMINALS?
5402    134B    DC  C1  17          CC   XPUTDC    ;YES - OUTPUT SI/SO
5403    134E     .   .   .   ;
5404    134E     .   .   .   ;  OUTPUT CHARACTERS FROM DISPLAY
5405    134E     .   .   .   ;
5406    134E     .   .   .   DSG020 EQU  $
5407    134E    CD  2C  24          CALL GETDSP    ;ANY CHARACTER?
5408    1351    DA  5D  13          JC   DSG100    ;NO - CHECK TERMINATION
5409    1354    CD  C1  17          CALL XPUTDC    ;YES - TRANSMIT THE CHARACTE
5410    1357    D2  4E  13          JNC  DSG020    ;CONTINUE IF NO DATA COMM ER
5411    135A    C3  B1  13          JMP  DSG230    ;ELSE, TERMINATE OUTPUT
5412    135D     .   .   .   ;
5413    135D     .   .   .   ;  NON-CHARACTER FOUND - CHECK TERMINATION
5414    135D     .   .   .   ;
5415    135D     .   .   .   DSG100 EQU  $
5416    135D    FA  8A  13          JM   DSG200    ;END OF DISPLAY - TERMINATE
5417    1360    47   .   .          MOV  B,A       ;SAVE EXIT STATUS
5418    1361    CD  59  10          CALL GTMODE    ;PAGE MODE ENABLED?
5419    1364    CA  99  13          JZ   DSG210    ;NO - END WITH CR(LF)
5420    1367    CD  7B  19          CALL CHKFMT    ;FORMAT MODE?
5421    136A    CA  76  13          JZ   DSG110    ;NO - SEND CR AND LF
5422    136D    3A  03  50          LDA  RECSEP    ;YES - END WITH RECORD
5423    1370    CD  C1  17          CALL XPUTDC      ;SEPARATOR
5424    1373    C3  46  13          JMP  DSG010    ;CONTINUE THRU DISPLAY
```

```
==========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
==========================================================================
 5426   1376   .    .    .    ;
 5427   1376   .    .    .    ;  EOL FOR NON-FORMAT PAGE MODE - SEND CR AND LF
 5428   1376   .    .    .    ;
 5429   1376   .    .    .    DSG110 EQU  $
 5430   1376   3E   0D   .           MVI  A,CR         ;SEND RETURN
 5431   1378   CD   C1   17          CALL XPUTDC         ;AND
 5432   137B   CD   0A   17          CALL SDTRM3           ;LINE FEED
 5433   137E   CD   8C   19          CALL CHKSFK       ;SOFT KEY DEFINE MODE?
 5434   1381   CC   6F   0A          CZ   LNFEED       ;NO - DO LINE FEED
 5435   1384   CD   9E   0F          CALL DISLN1       ;SET DISPLAY CURSOR ROW
 5436   1387   C3   46   13          JMP  DSG010       ;CONTINUE THRU DISPLAY
 5437   138A   .    .    .    ;
 5438   138A   .    .    .    ;  END OF DISPLAY - SEND TERMINATOR
 5439   138A   .    .    .    ;
 5440   138A   .    .    .    DSG200 EQU  $
 5441   138A   3A   04   50          LDA  BLKTRM       ;SEND BLOCK TERMINATOR
 5442   138D   CD   C1   17          CALL XPUTDC         ;CHARACTER
 5443   1390   CD   59   10          CALL GTMODE       ;PAGE MODE?
 5444   1393   CA   A7   13          JZ   DSG220       ;NO - SEND CR(LF)
 5445   1396   C3   AA   13          JMP  DSG225       ;YES - CLEAR XFR PENDING FLA
 5446   1399   .    .    .    ;******************************************
 5447   1399   .    .    .    ; NON-PAGE MODE TERMINATION - SEND CR(LF) *
 5448   1399   .    .    .    ;******************************************
 5449   1399   .    .    .    DSG210 EQU  $
 5450   1399   CD   76   19          CALL CHKFMS       ;FORMAT/SOFT KEY MODE?
 5451   139C   C2   A7   13          JNZ  DSG220       ;YES - DON'T DO LINE FEED
 5452   139F   3A   F3   FF          LDA  MDFLG2       ;NO - GET SOFT MODE FLAGS
 5453   13A2   E6   04   .           ANI  AUTOLF       ;AUTO LINE FEED ENABLED?
 5454   13A4   C4   6F   0A          CNZ  LNFEED       ;YES - DO LINE FEED
 5455   13A7   .    .    .    ;************************
 5456   13A7   .    .    .    ; SEND CR(LF) TERMINATOR *
 5457   13A7   .    .    .    ;************************
 5458   13A7   .    .    .    DSG220 EQU  $
 5459   13A7   CD   F6   16          CALL SDTRM1       ;SEND CR(LF)
 5460   13AA   .    .    .    DSG225 EQU  $
 5461   13AA   CD   20   12          CALL SDTER1       ;MARK END OF OUTPUT BLOCK
 5462   13AD   AF   .    .           XRA  A            ;RESET TERMINATOR FUNCTION
 5463   13AE   32   6D   FF          STA  TRMFCT         ;TO DELETE TERMINATOR
 5464   13B1   .    .    .    DSG230 EQU  $
 5465   13B1   01   FF   BF          LXI  B,-1-SENTER
 5466   13B4   CD   70   10          CALL CLBLXF       ;CLEAR ENTER PENDING FLAG
 5467   13B7   C3   23   20          JMP  CRADV1       ;CLEAR CURSOR ADVANCE FLAG
 5468   13BA   .    .    .    ;                           AND EXIT
 5470   13BA    .   .   .   ;***************************************
 5471   13BA    .   .   .   ; A2OUTB - PUT BYTE INTO OUTPUT BUFFER *
 5472   13BA    .   .   .   ;***************************************
 5473   13BA    .   .   .   ;  ENTRY:  A = BYTE TO BE OUTPUT
 5474   13BA    .   .   .   ;
 5475   13BA    .   .   .   ;  EXIT :  H = BASEH
 5476   13BA    .   .   .   ;          B2DEND = B2DEND + 1
 5477   13BA    .   .   .   ;          D,E,L DESTROYED
 5478   13BA    .   .   .   ;
 5479   13BA    .   .   .   ;  ECOUTB - OUTPUT <ESC>
 5480   13BA    .   .   .   ;
 5481   13BA    .   .   .   ;      ENTRY:  B = SECOND CHARACTER IN ESCAPE SEQ
 5482   13BA    .   .   .   ;
 5483   13BA    .   .   .   ECOUTB EQU  $
 5484   13BA    3E  1B  .          MVI  A,ESC      ;SET A TO ESC
 5485   13BC    CD  C0  13         CALL A2OUTB     ;PUT ESC INTO OUTPUT BUFFER
 5486   13BF    .   .   .   B2OUTB EQU  $
 5487   13BF    78  .   .          MOV  A,B        ;PUT SECOND CHAR INTO A-REG
 5488   13C0    .   .   .   ;                      FALL INTO OUTPUT ROUTINE
 5489   13C0    .   .   .   ;
 5490   13C0    .   .   .   A2OUTB EQU  $
 5491   13C0    21  3B  FF         LXI  H,B2DEND
 5492   13C3    34  .   .          INR  M          ;INCREMENT TO NEXT POSITION
 5493   13C4    6E  .   .          MOV  L,M        ;GET NEW ADDRESS
 5494   13C5    77  .   .          MOV  M,A        ;STORE THE BYTE
 5495   13C6    C9  .   .          RET             ;RETURN
```

```
ITEM    LOC    OBJECT CODE   SOURCE STATEMENTS
5497    13C7   .    .    .   ;*****************************
5498    13C7   .    .    .   ; ENTLCL - ENTER LOCAL MODE *
5499    13C7   .    .    .   ;*****************************
5500    13C7   .    .    .   ENTLCL EQU  $
5501    13C7   CD   4D   10         CALL CKEDIT      ;EDIT MODE ENABLED?
5502    13CA   CA   D5   13         JZ   ENL100      ;NO - GO INTO LOCAL MODE
5503    13CD   3E   08   .          MVI  A,REMOTE    ;YES - INHIBIT TRANSITION TO
5504    13CF   21   F3   FF         LXI  H,MDFLG2      ;LOCAL MODE
5505    13D2   B6   .    .          ORA  M           ;FORCE REMOTE FLAG ON
5506    13D3   77   .    .          MOV  M,A
5507    13D4   C9   .    .          RET              ;RETURN
5508    13D5   .    .    .   ;
5509    13D5   .    .    .   ENL100 EQU  $
5510    13D5   3E   04   .          MVI  A,SETLCL    ;SET DATACOM FOR LOCAL
5511    13D7   CD   42   12         CALL DCMCTL         ;OPERATION
5512    13DA   3E   EF   .          MVI  A,377Q-REMSET  ;CLEAR REMOTE MODE FLAG
5513    13DC   .    .    .   ;******************************
5514    13DC   .    .    .   ; CLCMFL - CLEAR COMMON FLAGS *
5515    13DC   .    .    .   ;******************************
5516    13DC   .    .    .   ;
5517    13DC   .    .    .   ;  ENTRY:  A = 377B-FLAG BIT TO BE CLEARED
5518    13DC   .    .    .   ;
5519    13DC   .    .    .   ;  EXIT :  A,H,L DESTROYED
5520    13DC   .    .    .   ;
5521    13DC   .    .    .   CLCMFL EQU  $
5522    13DC   21   F8   FF         LXI  H,CMFLGS
5523    13DF   A6   .    .          ANA  M           ;CLEAR THE FLAG BIT
5524    13E0   77   .    .          MOV  M,A         ;STORE THE NEW SETTINGS
5525    13E1   C9   .    .          RET              ;RETURN
```

```
5527   13E2   .   .   .    ;
5528   13E2   .   .   .    ;  ENTREM - ENTER REMOTE MODE
5529   13E2   .   .   .    ;
5530   13E2   .   .   .    ENTREM EQU  $
5531   13E2   CD  4D  10          CALL CKEDIT    ;EDIT MODE ENABLED?
5532   13E5   CA  F0  13          JZ   ENR100    ;NO - GO INTO REMOTE MODE
5533   13E8   3E  F7  .           MVI  A,377Q-REMOTE  ;YES - INHIBIT
5534   13EA   21  F3  FF          LXI  H,MDFLG2    ;TRANSITION TO REMOTE MODE
5535   13ED   A6  .   .           ANA  M           ;FORCE REMOTE FLAG OFF
5536   13EE   77  .   .           MOV  M,A
5537   13EF   C9  .   .           RET              ;RETURN
5538   13F0   .   .   .    ;
5539   13F0   .   .   .    ENR100 EQU  $
5540   13F0   97  .   .           SUB  A           ;CLEAR THE DATA PENDING
5541   13F1   32  70  FF          STA  MFLGS           ;FLAGS
5542   13F4   3E  FC  .           MVI  A,377Q-SBINRY-SDVREC
5543   13F6   CD  AA  04          CALL CLRMF2    ;CLEAR BINARY RECORD PENDING
5544   13F9   3E  03  .           MVI  A,SETREM  ;SET DATACOM FOR REMOTE
5545   13FB   CD  4D  12          CALL DCMCT1        ;OPERATION
5546   13FE   3E  10  .           MVI  A,REMSET  ;SET REMOTE MODE FLAG
5547   1400   .   .   .    ;****************************
5548   1400   .   .   .    ; STCMFL - SET COMMON FLAGS *
5549   1400   .   .   .    ;****************************
5550   1400   .   .   .    ;
5551   1400   .   .   .    ;  ENTRY:  A = FLAG BIT TO BE SET
5552   1400   .   .   .    ;
5553   1400   .   .   .    ;  EXIT :  A,H,L DESTROYED
5554   1400   .   .   .    ;
5555   1400   .   .   .    STCMFL EQU  $
5556   1400   21  F8  FF          LXI  H,CMFLGS
5557   1403   B6  .   .           ORA  M           ;ADD BIT TO "CMFLGS"
5558   1404   77  .   .           MOV  M,A         ;STORE NEW SETTINGS
5559   1405   C9  .   .           RET              ;RETURN
```

```
 5561   1406    .    .    .   ;
 5562   1406    .    .    .   ; * * * * * * * * * * * * * * * * * * * * * * * *
 5563   1406    .    .    .   ;
 5564   1406    .    .    .   ;  FCTKEY - FUNCTION KEY PRESSED (F1-F8)
 5565   1406    .    .    .   ;
 5566   1406    .    .    .   ;      ENTRY:  C = FUNCTION KEY CODE (360-367B)
 5567   1406    .    .    .   ;
 5568   1406    .    .    .   ;      EXIT :  DFLGS(FCTK2D) = 1, FUNCTION KEY
 5569   1406    .    .    .   ;                 DATA TO BE USED AS NORMAL
 5570   1406    .    .    .   ;                 KEYBOARD CHARACTERS
 5571   1406    .    .    .   ;               DFLGS(FCTK2D) = 0
 5572   1406    .    .    .   ;                 MFLGS(SFCTKY) = 0, KEY WAS
 5573   1406    .    .    .   ;                   INTERPRETED LOCALLY ONLY
 5574   1406    .    .    .   ;                 MFLGS(SFCTKY) = 1, DATA WAITING
 5575   1406    .    .    .   ;                   FOR BLOCK TRANSFER TRIGGER TO
 5576   1406    .    .    .   ;                   SEND TO CPU
 5577   1406    .    .    .   ;               ALL REGISTERS DESTROYED
 5578   1406    .    .    .   ;
 5579   1406    .    .    .   FCTKEY EQU  $
 5580   1406   79    .    .         MOV  A,C       ;COMPUTE NUMBER OF LINES TO
 5581   1407   87    .    .         ADD  A           ;SEARCH:
 5582   1408   D6   DF    .         SUI  FCTADJ        ;2*(FUNCTION NUMBER) - 1
 5583   140A   21   A6   FF         LXI  H,SFTKYS
 5584   140D   CD   F6   0A         CALL MLKSC1    ;LOCATE THE ATTRIBUTE LINE
 5585   1410    .    .    .   ;
 5586   1410    .    .    .   ;  DEFINITION FOUND - PERFORM FUNCTION
 5587   1410    .    .    .   ;
 5588   1410   7D    .    .         MOV  A,L       ;COMPUTE LOCATION OF
 5589   1411   D6   08    .         SUI  ATBLOC      ;ATTRIBUTE CODE
 5590   1413   5F    .    .         MOV  E,A
 5591   1414   54    .    .         MOV  D,H
 5592   1415   CD   6D   19         CALL CHAIN     ;GET ADDRESS OF DATA LINE
 5593   1418   22   A4   FF         SHLD CURFKY    ;SAVE FIRST CHARACTER ADDRES
 5594   141B    .    .    .   ;                      TO FORCE SKIP OVER "ENDPR"
 5595   141B   1A    .    .         LDAX D         ;GET ATTRIBUTE CODE
 5596   141C   FE   4E    .         CPI  N         ;NORMAL MODE?
 5597   141E   DA   32   14         JC   FCT200    ;< - DO LOCAL ONLY
 5598   1421   3E   10    .         MVI  A,FCTK2D    ;(SET DATA XFR FLAG)
 5599   1423   CA   11   17         JZ   SETDFL    ;YES - SET NORMAL KEY XFR
 5600   1426   CD   59   10         CALL GTMODE    ;> - SET BLOCK TRANSFER
 5601   1429   01   00   20         LXI  B,SFCTKY    ;FOR FUNCTION KEY
 5602   142C   CA   CD   16         JZ   SBLXFA    ;SET FLAG FOR NOT PAGE MODE
 5603   142F   C3   D5   16         JMP  SBLXF1      ;ELSE SET FOR PAGE XFR
```

```
5605    1432   .   .   .    ;*********************************
5606    1432   .   .   .    ; PERFORM LOCAL ONLY KEY FUNCTION *
5607    1432   .   .   .    ;*********************************
5608    1432   .   .   .    FCT200 EQU  $
5609    1432   CD  42  15          CALL GTFCTK      ;GET FUNCTION KEY DATA
5610    1435   C8  .   .           RZ               ;NONE LEFT - RETURN
5611    1436   .   .   .    FCT210 EQU  $
5612    1436   32  9C  FF          STA  CHARIN      ;SAVE FUNCTION KEY CHARACTER
5613    1439   CD  50  03          CALL CHINT       ;PROCESS DATA LOCALLY
5614    143C   3A  9C  FF          LDA  CHARIN      ;RECALL FUNCTION CHARACTER
5615    143F   FE  0D  .           CPI  CR          ;IS IT A RETURN?
5616    1441   C2  32  14          JNZ  FCT200      ;NO - DO THE NEXT BYTE
5617    1444   3A  F3  FF          LDA  MDFLG2      ;YES - GET HARD MODE FLAGS
5618    1447   E6  04  .           ANI  AUTOLF      ;AUTO LINE FEED ENABLED?
5619    1449   CA  32  14          JZ   FCT200      ;NO - DO NEXT FUNCTION BYTE
5620    144C   0E  0A  .           MVI  C,LF        ;YES - PERFORM LINE FEED
5621    144E   C3  36  14          JMP  FCT210        ;FUNCTION
```

```
 5623   1451   .   .   .   ;
 5624   1451   .   .   .   ; * * * * * * * * * * * * * * * * * * * * * * *
 5625   1451   .   .   .   ;
 5626   1451   .   .   .   ;  FKEYGO - TRANSMIT FUNCTION KEY
 5627   1451   .   .   .   ;
 5628   1451   .   .   .   ;      ENTRY:  DON'T CARE
 5629   1451   .   .   .   ;
 5630   1451   .   .   .   ;      EXIT :  MFLGS1(SFCTKY) = 0
 5631   1451   .   .   .   ;              ALL REGISTERS DESTROYED
 5632   1451   .   .   .   ;
 5633   1451   .   .   .   FKEYGO EQU  $
 5634   1451   01  FF  DF         LXI  B,-1-SFCTKY  ;CLEAR FUNCTION KEY
 5635   1454   CD  70  10         CALL CLBLXF       ;PENDING FLAG
 5636   1457   3A  6E  FF         LDA  DFLGS       ;GET DATA TRANSFER FLAGS
 5637   145A   E6  10  .          ANI  FCTK2D      ;OPERATE AS NORMAL KEY?
 5638   145C   C0  .   .          RNZ              ;YES - RETURN TO WAIT LOOP
 5639   145D   .   .   .   ;
 5640   145D   .   .   .   ;  TRANSMIT FUNCTION KEY DATA
 5641   145D   .   .   .   ;
 5642   145D   .   .   .   FKG010 EQU  $
 5643   145D   CD  42  15         CALL GTFCTK      ;GET NEXT FUNCTION KEY CHAR
 5644   1460   CA  1D  12         JZ   SDTERM      ;SEND TERMINATOR IF NO MORE
 5645   1463   .   .   .   ;                         DATA
 5646   1463   21  04  50         LXI  H,BLKTRM
 5647   1466   BE  .   .          CMP  M           ;BLOCK TERMINATOR CHARACTER?
 5648   1467   CA  1D  12         JZ   SDTERM      ;YES - OUTPUT TERMINATOR
 5649   146A   CD  C1  17         CALL XPUTDC      ;NORMAL DATA - TRANSMIT IT
 5650   146D   C3  5D  14         JMP  FKG010      ;DO NEXT CHARACTER
```

```
=========================================================================
5652   1470   .   .   .    ;
5653   1470   .   .   .    ;  MNMDON - MONITOR MODE ON
5654   1470   .   .   .    ;
5655   1470   .   .   .    MNMDON EQU  $
5656   1470   3E  08  .           MVI  A,SETMON  ;SET DATACOM MONITOR
5657   1472   CD  42  12          CALL DCMCTL       ;MODE
5658   1475   C0  .   .           RNZ            ;DON'T MONITOR IF NOT SET
5659   1476   06  FF  .           MVI  B,377Q    ;SET TO BLINK LED
5660   1478   C3  7D  14          JMP  FD0100    ;SET FUNCTION TABLE
5661   147B   .   .   .    ;****************************************
5662   147B   .   .   .    ; FDISON - TURN ON FUNCTION DISABLE MODE *
5663   147B   .   .   .    ;****************************************
5664   147B   .   .   .    FDISON EQU $
5665   147B   06  00  .           MVI  B,0       ;SET FOR NO BLINK
5666   147D   .   .   .    FD0100 EQU  $
5667   147D   3E  01  .           MVI  A,DSPFNC  ;TURN ON DISPLAY FUNCTIONS
5668   147F   CD  0E  48          CALL ZSTMD1       ;FLAG
5669   1482   21  8C  27          LXI  H,FDISTB  ;SET H,L TO NEW RANGE TABLE
5670   1485   C3  A1  04          JMP  ESCEN1    ;SET RANGE TABLE AND EXIT
5671   1488   .   .   .    ;************************************
5672   1488   .   .   .    ; FDISOF - TURN OFF FUNCTION DISABLE *
5673   1488   .   .   .    ;************************************
5674   1488   .   .   .    FDISOF EQU  $
5675   1488   CD  8C  19          CALL CHKSFK    ;SOFT KEY DEFINE MODE?
5676   148B   CA  94  14          JZ   FOF010    ;NO - DO NORMAL PROCESSING
5677   148E   CD  A6  12          CALL DCXB2D    ;INPUT FROM KEYBOARD?
5678   1491   C4  8D  0C          CNZ  SFKYOF    ;NO - RESTURE NORMAL DISPLAY
5679   1494   .   .   .    FOF010 EQU  $
5680   1494   CD  B6  14          CALL FDESC1    ;DISPLAY INPUT CHARACTER
5681   1497   3A  69  FF          LDA  LCHAR
5682   149A   FE  1B  .           CPI  ESC       ;WAS THE LAST CHAR <ESC>?
5683   149C   C0  .   .           RNZ          ;NO - RETURN
5684   149D   .   .   .    ;                     YES - TURN OFF DISPLAY FCTS
5685   149D   .   .   .    DFCTOF EQU  $
5686   149D   3E  09  .           MVI  A,SETNRM  ;RESTORE DATACOM TO
5687   149F   CD  11  50          CALL ZDCCTL       ;NORMAL MODE
5688   14A2   CD  95  04          CALL ESCEND  ;YES - TURN OFF DISABLE MODE
5689   14A5   3E  01  .           MVI  A,DSPFNC  ;TURN OFF DISPLAY FUNCTIONS
5690   14A7   C3  11  48          JMP  ZCLMD1       ;FLAG
5691   14AA   .   .   .    ;**************************
5692   14AA   .   .   .    ; FUNCTION DISABLE ESCAPE *
5693   14AA   .   .   .    ;**************************
5694   14AA   .   .   .    FDESC  EQU  $
5695   14AA   CD  8C  19          CALL CHKSFK    ;SOFT KEY DEFINE MODE?
5696   14AD   CA  B6  14          JZ   FDESC1    ;NO - DO NORMAL PROCESSING
5697   14B0   CD  A6  12          CALL DCXB2D    ;INPUT FROM KEYBOARD?
5698   14B3   C4  8D  0C          CNZ  SFKYOF    ;NO - RESTORE NORMAL DISPLAY
5699   14B6   .   .   .    FDESC1 EQU  $
5700   14B6   CD  1A  23          CALL DSPCHR    ;DISPLAY THE ESCAPE CODE
5701   14B9   C3  23  20          JMP  CRADV1    ;RESET CURSOR ADVANCE FLAG T
5702   14BC   .   .   .    ;                       FORCE ANALYSIS OF NEXT
5703   14BC   .   .   .    ;                       INPUT CHARACTER FOR Z
```

```
 5705   14BC   .    .    .    ;*********************************************
 5706   14BC   .    .    .    ; FUNCTION TABLE FOR TERMINAL FUNCTION KEYS *
 5707   14BC   .    .    .    ;*********************************************
 5708   14BC   .    .    .    FNCTAB EQU  $
 5709   14BC   D8   12   .           DW   DPSEND     ;230 - ENTER KEY
 5710   14BE   3B   12   .           DW   BRKDC      ;231 - BREAK KEY
 5711   14C0   9D   14   .           DW   DFCTOF     ;232 - DISPLAY FUNCTIONS OFF
 5712   14C2   8C   15   .           DW   IOKEYS     ;233 - I/O CONTROL KEY
 5713   14C4   8C   15   .           DW   IOKEYS     ;234 - READ KEY
 5714   14C6   8C   15   .           DW   IOKEYS     ;235 - RECORD KEY
 5715   14C8   8C   15   .           DW   IOKEYS     ;236 - SELECT KEY
 5716   14CA   8C   15   .           DW   IOKEYS     ;237 - CONDITION TAPES
 5717   14CC   8C   15   .           DW   IOKEYS     ;240 - (CONTROL) READ KEY
 5718   14CE   .    .    .    ;
 5719   0098   .    .    .    ENTRCD EQU  230Q       ;ENTER KEY CODE
 5720   009D   .    .    .    RCKYCD EQU  235Q       ;RECORD KEY CODE
 5721   009E   .    .    .    SLKYCD EQU  236Q       ;SELECT KEY CODE
 5722   00A0   .    .    .    CTRDKY EQU  240Q       ;CONTROL READ KEY CODE
 5723   0098   .    .    .    FNCLWR EQU  230Q       ;FUNCTION CODE LOWER LIMIT
 5724   00A1   .    .    .    FNCLIM EQU  241Q       ;FUNCTION CODE UPPER LIMIT
 5725   14CE   .    .    .    ;
 5726   008E   .    .    .    ESCSO  EQU  216Q       ;<ESC>-<SO> CODE
 5727   00E4   .    .    .    ESCLWD EQU  344Q       ;<ESC>-<LOWER CASE D> CODE
 5728   00F0   .    .    .    F1CODE EQU  360Q       ;F1 CODE
 5729   00F7   .    .    .    F8CODE EQU  367Q       ;F8 CODE
 5730   01E0   .    .    .    FCTAD1 EQU  F1CODE*2   ;FUNCTION CODE ADJUSTMENT
 5731   FFDF   .    .    .    FCTADJ EQU  -FCTAD1/256*256+FCTAD1-1  ;FACTOR
 5732   00FD   .    .    .    STFOR2 EQU  375Q       ;SET FOREIGN MODE STEP 2
 5733   00FE   .    .    .    STFOR1 EQU  376Q       ;SET FOREIGN MODE STEP 1
 5734   00FF   .    .    .    ENHNCF EQU  377Q       ;ENHANCE DISPLAY FUNCTION
 5735   14CE   .    .    .    ;*********************************
 5736   14CE   .    .    .    ; FUNCTION ADDRESSES FOR I/O KEYS *
 5737   14CE   .    .    .    ;*********************************
 5738   14CE   .    .    .    IOKYTB EQU  $
 5739   14CE   02   28   .           DW   IOCKEY     ;I/O CONTROL KEY
 5740   14D0   05   28   .           DW   REDKEY     ;READ KEY
 5741   14D2   0B   28   .           DW   RECKEY     ;RECORD KEY
 5742   14D4   0E   28   .           DW   SELKEY     ;SELECT KEY
 5743   14D6   14   28   .           DW   CONDTN     ;CONDITION TAPES
 5744   14D8   08   28   .           DW   CTLRED     ;(CONTROL) READ KEY
```

```
5746    14DA   .    .    .    ;
5747    14DA   .    .    .    ;  DISPLAY STRINGS FOR SOFT KEY DISPLAY
5748    14DA   .    .    .    ;
5749    14DA   .    .    .    ATBLIN EQU   $
5750    14DA   CC   20   1B          DB    EOL,ABLNK,ESC,ENDPR
5751    14DE   .    .    .    ;
5752    14DE   CC   C0   .           DB    EOL,STPR
5753    14E0   .    .    .    ATB010 EQU   $
5754    14E0   54   C8   C1          DB    'T',SFKYAT,ENDPR,' '
5755    14E4   80   30   66          DB    NORMAL,'0',146Q,INVRS,0
5756    0008   .    .    .    ATBLOC EQU   $-ATB010-1  ;ATTRIBUTE LOCATION IN BLK
5757    000E   .    .    .    ATBLEN EQU   $-ATBLIN-1  ;ATTRIBUTE LINE LENGTH
5758    0002   .    .    .    CHRLOC EQU   2           ;CHARACTER LOCATION IN STRIN
```

```
 5760   14E9   .   .   .   ;*********************************************
 5761   14E9   .   .   .   ; FNDTAB - FIND TAB MASK                        *
 5762   14E9   .   .   .   ; EXIT: L,H = ADDR OF BYTE CONTAINING TAB BIT *
 5763   14E9   .   .   .   ;          A = MASK FOR TAB BIT                 *
 5764   14E9   .   .   .   ;*********************************************
 5765   14E9   .   .   .   FNDTAB EQU $
 5766   14E9   3A  C1  FF         LDA  CURCOL   ;GET CURSOR COLUMN
 5767   14EC   47  .   .          MOV B,A       ;SAVE IN B
 5768   14ED   .   .   .   FNDTB1 EQU $
 5769   14ED   E6  F8  .          ANI 370Q      ;MASK OFF 3 LSB'S
 5770   14EF   0F  .   .          RRC           ;RIGHT-ADJUST MSB'S
 5771   14F0   0F  .   .          RRC
 5772   14F1   0F  .   .          RRC
 5773   14F2   C6  78  .          ADI HTBTBL    ;ADD BASE OF TAB TABLE
 5774   14F4   6F  .   .          MOV L,A       ;SAVE IN L
 5775   14F5   78  .   .          MOV A,B       ;GET CURSOR COLUMN
 5776   14F6   E6  07  .          ANI 7         ;GET 3 LSB'S
 5777   14F8   47  .   .          MOV B,A       ;SAVE IN B
 5778   14F9   04  .   .          INR B         ;ADJUST BIT NUMBER
 5779   14FA   .   .   .   ;********************
 5780   14FA   .   .   .   ; FNDTB2 - SET BIT N *
 5781   14FA   .   .   .   ;********************
 5782   14FA   .   .   .   ;
 5783   14FA   .   .   .   ;   ENTRY:  B = BIT NUMBER TO BE SET
 5784   14FA   .   .   .   ;
 5785   14FA   .   .   .   ;   EXIT :  A = BYTE WITH BIT N SET
 5786   14FA   .   .   .   ;           B = 0
 5787   14FA   .   .   .   ;
 5788   14FA   .   .   .   FNDTB2 EQU  $
 5789   14FA   3E  80  .          MVI A,200Q    ;SET BIT 7 OF A
 5790   14FC   .   .   .   FTB100 EQU $
 5791   14FC   07  .   .          RLC           ;SHIFT LEFT 1 POSITION
 5792   14FD   05  .   .          DCR B         ;DECREMENT BIT COUNT
 5793   14FE   C2  FC  14         JNZ FTB100    ;CONTINUE IF NOT DONE
 5794   1501   C9  .   .          RET           ;RETURN
```

```
5796    1502    .    .    .     ;*******************
5797    1502    .    .    .     ; EXIT FORMAT MODE *
5798    1502    .    .    .     ;*******************
5799    1502    .    .    .     FORMOF EQU $
5800    1502    3E   08   .            MVI   A,FORMAT  ;SET BIT TO BE CLEAPED
5801    1504    32   C2   FF           STA   PROFLD    ;SET PROTECT FLAG FOR UNPROT
5802    1507    C3   11   48           JMP   ZCLMD1    ;CLEAR FORMAT MODE FLAG
```

```
150A   .   .   .   ;*********************************************
150A   .   .   .   ; FRECNT - CHECK THE NUMBER OF FREE BLOCKS    *
150A   .   .   .   ;                                             *
150A   .   .   .   ;          EXIT: Z=F, NOT ENOUGH FREE BLOCKS *
150A   .   .   .   ;                 Z=T, ENOUGH FREE BLOCKS     *
150A   .   .   .   ;*********************************************
150A   .   .   .   FRECNT EQU $
150A   06  19  .          MVI  B,25       ;SET DESIRED NUMBER OF BLOCK
150C   11  AA  FF         LXI  D,FRBLKS-2  ;SET TO FREE LIST HEAD
150F   .   .   .   FRC010 EQU  $
150F   EB  .   .          XCHG            ;SET H,L TO ADDRESS OF LSB
1510   23  .   .          INX  H            ;PART OF PREVIOUS LINE
1511   23  .   .          INX  H            ;POINTER
1512   7E  .   .          MOV  A,M        ;GET LSB OF NEXT LINE LINK
1513   B7  .   .          ORA  A          ;ANY MORE FREE BLOCKS?
1514   CA  30  15         JZ   FRC100     ;NO - TRY TO GET MORE
1517   05  .   .          DCP  B          ;ENOUGH FREE BLOCKS?
1518   C8  .   .          RZ              ;YES - RETURN SUCCESSFUL
1519   CD  6D  19         CALL CHAIN      ;NO - GET NEXT LINE ADDRESS
151C   54  .   .          MOV  D,H        ;SAVE NEXT LINE ADDRESS IN
151D   5D  .   .          MOV  E,L          ;D,E
151E   .   .   .   FRC050 EQU  $
151E   E6  F0  .          ANI  360Q       ;COMPUTE ADDRESS OF NEXT
1520   6F  .   .          MOV  L,A          ;BLOCK LINK
1521   7E  .   .          MOV  A,M        ;GET THE LSB OF THE LINK
1522   2F  .   .          CMA  ;A          IS IT AN EOL LINK (LOWER
1523   E6  0F  .          ANI  BLKSM        ;FOUR BITS NOT ALL ONES)?
1525   C2  0F  15         JNZ  FRC010     ;NO - GO TO THE NEXT LINE
1528   05  .   .          DCR  B          ;ENOUGH FREE BLOCKS FOUND?
1529   C8  .   .          RZ              ;YES - RETURN SUCCESSFUL
152A   CD  6D  19         CALL CHAIN      ;NO - GO TO THE NEXT BLOCK
152D   C3  1E  15         JMP  FRC050     ;CHECK FOR END OF LINE
1530   .   .   .   ;********************************************
1530   .   .   .   ; NOT ENOUGH FREE BLOCKS - TRY TO GET MORE *
1530   .   .   .   ;********************************************
1530   .   .   .   FRC100 EQU  $
1530   CD  13  06         CALL PTBLK      ;REMOVE A LINE FROM DISPLAY
1533   C2  0A  15         JNZ  FRECNT     ;RECOUNT IF LINE FREED
1536   3C  .   .          INR  A            ;(FORCE NZ)
1537   C9  .   .          RET             ;RETURN FAIL OTHERWISE
```

```
============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                      PAGE 169
============================================================================
 5845   1538    .   .   .   ;**********************************************
 5846   1538    .   .   .   ; FRNCT1 - FOREIGN MODE CONTROL 1 (<ESC>-"<") *
 5847   1538    .   .   .   ;**********************************************
 5848   1538    .   .   .   FRNCT1 EQU  $
 5849   1538   3E  0E   .          MVI  A,FRNMD1  ;SET KEYBOARD FOREIGN MODE 1
 5850   153A   C3  08  48          JMP  ZKBCTL
 5851   153D    .   .   .   ;**********************************************
 5852   153D    .   .   .   ; FRNCT2 - FOREIGN MODE CONTROL 2 (<ESC>-">") *
 5853   153D    .   .   .   ;**********************************************
 5854   153D    .   .   .   FRNCT2 EQU  $
 5855   153D   3E  0F   .          MVI  A,FRNMD2  ;SET KEYBOARD FOREIGN MODE 2
 5856   153F   C3  08  48          JMP  ZKBCTL
```

```
================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                        PAGE 170
================================================================================
 5858   1542   .    .    .   ;
 5859   1542   .    .    .   ; * * * * * * * * * * * * * * * * * * * * * * * *
 5860   1542   .    .    .   ;
 5861   1542   .    .    .   ;  GTFCTK - GET FUNCTION KEY
 5862   1542   .    .    .   ;
 5863   1542   .    .    .   ;        ENTRY:  DON'T CARE
 5864   1542   .    .    .   ;
 5865   1542   .    .    .   ;        EXIT :  NZ - FUNCTION KEY CHAR AVAILABLE
 5866   1542   .    .    .   ;                  A = C = FUNCTION KEY CHARACTER
 5867   1542   .    .    .   ;                Z - NO FUNCTION KEY CHAR AVAILABLE
 5868   1542   .    .    .   ;                  DFLGS(FCTK2D) = 0
 5869   1542   .    .    .   ;                  A DESTROYED
 5870   1542   .    .    .   ;                D-L DESTROYED
 5871   1542   .    .    .   ;
 5872   1542   .    .    .   GTFCTK EQU  $
 5873   1542   2A   A4   FF         LHLD CURFKY     ;GET LAST FUNCTION KEY
 5874   1545   .    .    .   ;                        CHARACTER ADDRESS
 5875   1545   CD   86   0B         CALL NXTCHO     ;GET THE NEXT CHARACTER
 5876   1548   C2   53   15         JNZ  GTF010     ;EOL LINK - DO EOL EXIT
 5877   154B   EB   .    .          XCHG
 5878   154C   22   A4   FF         SHLD CURFKY     ;STORE NEW ADDRESS
 5879   154F   FE   80   .          CPI  ADEL+1     ;IS CHARACTER ASCII?
 5880   1551   4F   .    .          MOV  C,A          ;(PUT DATA IN C-REGISTER)
 5881   1552   F8   .    .          RM              ;YES - RETURN
 5882   1553   .    .    .   ;
 5883   1553   .    .    .   ;  EOL FOUND - CLEAR FCTK2D FLAG        .
 5884   1553   .    .    .   ;
 5885   1553   .    .    .   GTF010 EQU  $
 5886   1553   3E   EF   .          MVI  A,377Q-FCTK2D
 5887   1555   CD   01   16         CALL CLRDFL     ;CLEAR FLAG FROM FLAG WORD
 5888   1558   BF   .    .          CMP  A          ;SET Z TRUE
 5889   1559   C9   .    .          RET             ;RETURN
```

```
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
5891    155A   .    .    .   ;***************************
5892    155A   .    .    .   ; HTBSET - TAB SET ROUTINE *
5893    155A   .    .    .   ;***************************
5894    155A   .    .    .   HTBSET EQU $
5895    155A   CD   E9   14         CALL FNDTAB  ;GET TABLE ENTRY FOR COLUMN
5896    155D   B6   .    .          ORA M        ;SET TAB
5897    155E   77   .    .          MOV M,A
5898    155F   C9   .    .          RET          ;RETURN
5899    1560   .    .    .   ;*****************************
5900    1560   .    .    .   ; HTBCLR - TAB CLEAR ROUTINE *
5901    1560   .    .    .   ;*****************************
5902    1560   .    .    .   HTBCLR EQU $
5903    1560   CD   E9   14         CALL FNDTAB  ;GET TABLE ENTRY FOR COLUMN
5904    1563   EE   FF   .          XRI 377Q     ;COMPLEMENT MASK
5905    1565   A6   .    .          ANA M        ;CLEAR TAB
5906    1566   77   .    .          MOV M,A
5907    1567   C9   .    .          RET          ;RETURN
```

```
5909   1568    .   .   .    ;**********************************************
5910   1568    .   .   .    ; IOBNGO - FAST BINARY READ ESCAPE SEQUENCE *
5911   1568    .   .   .    ;**********************************************
5912   1568    .   .   .    IOBNGO EQU  $
5913   1568    21  2C  28          LXI  H,CTDCDP  ;EXECUTE FAST BINARY READ
5914   156B    C3  93  15          JMP  IORMGO       ;IF I/O ROM PRESENT
5915   156E    .   .   .    ;*****************************
5916   156E    .   .   .    ; IOBSYC - WAIT FOR CTU IDLE *
5917   156E    .   .   .    ;*****************************
5918   156E    .   .   .    IOBSYC EQU  $
5919   156E    21  3A  28          LXI  H,BSYCHK  ;GO TO CTU BUSY CHECK
5920   1571    CD  93  15          CALL IORMGO       ;ROUTINE
5921   1574    3A  55  FF          LDA  CMND      ;GET CURRENT CTU COMMAND
5922   1577    E6  01  .           ANI  RUN       ;TAPE STILL RUNNING?
5923   1579    C8  .   .           RZ             ;NO - RETURN
5924   157A    32  4F  FF          STA  IOCERR    ;YES - CLEAR "IOCERR"
5925   157D    C3  6E  15          JMP  IOBSYC    ;CONTINUE WAITING
5926   1580    .   .   .    ;***************************************
5927   1580    .   .   .    ; IOCTGO - I/O CONTROL ESCAPE SEQUENCE *
5928   1580    .   .   .    ;***************************************
5929   1580    .   .   .    IOCTGO EQU  $
5930   1580    21  1A  28          LXI  H,IOCNTL  ;EXECUTE I/O CONTROL ESCAPE
5931   1583    C3  93  15          JMP  IORMGO       ;SEQ IF I/O ROM PRESENT
5932   1586    .   .   .    ;***********************************
5933   1586    .   .   .    ; IOCTMN - MONITOR CARTRIDGE TAPES *
5934   1586    .   .   .    ;***********************************
5935   1586    .   .   .    IOCTMN EQU  $
5936   1586    21  2F  28          LXI  H,CTMON   ;GET MONITOR ADDRESS
5937   1589    C3  93  15          JMP  IORMGO    ;EXECUTE IF CODE PRESENT
5938   158C    .   .   .    ;***********************
5939   158C    .   .   .    ; IOKEYS - I/O KEY HIT *
5940   158C    .   .   .    ;***********************
5941   158C    .   .   .    ;
5942   158C    .   .   .    ;  ENTRY:  D,E = KEY INDEX
5943   158C    .   .   .    ;
5944   158C    .   .   .    IOKEYS EQU  $
5945   158C    21  C8  14          LXI  H,IOKYTB-6
5946   158F    19  .   .           DAD  D          ;COMPUTE KEY FUNCTION ADDRES
5947   1590    CD  6D  19          CALL CHAIN      ;EXECUTE KEY FUNCTION IF I/O
5948   1593    .   .   .    ;                        ROM PRESENT
```

```
5950   1593    .    .    .   ;*******************************************
5951   1593    .    .    .   ; IORMGO - PERFORM FUNCTION IF OPTION ROMS *
5952   1593    .    .    .   ;   ARE PRESENT                             *
5953   1593    .    .    .   ;*******************************************
5954   1593    .    .    .   ;
5955   1593    .    .    .   ;  ENTRY:  H,L = VECTOR TO BE ENTERED
5956   1593    .    .    .   ;
5957   1593    .    .    .   ;  EXIT :  NC - FUNCTION EXECUTED
5958   1593    .    .    .   ;               REGISTERS SET ACCORDING TO FUNCTION
5959   1593    .    .    .   ;           C - FUNCTION NOT EXECUTED
5960   1593    .    .    .   ;               A DESTROYED
5961   1593    .    .    .   ;
5962   1593    .    .    .   IORMGO EQU  $
5963   1593    E5   .    .          PUSH H          ;PUT FUNCTION ADDR ON STACK
5964   1594    2E   00   .          MVI  L,0        ;CHECK ROM START LOCATION
5965   1596    CD   A3   15         CALL IORMG1     ;DOES ROM EXIST?
5966   1599    C8   .    .          RZ              ;YES - EXECUTE FUNCTION
5967   159A    21   59   0F         LXI  H,NODRVR   ;NO - SET ERROR MESSAGE TO
5968   159D    22   F1   FF         SHLD MSGPT1       ;"NO DEVICE DRIVER"
5969   15A0    E1   .    .          POP  H          ;RESTORE STACK
5970   15A1    37   .    .          STC             ;RETURN FUNCTION NOT
5971   15A2    C9   .    .          RET               ;EXECUTED (C-TRUE)
5972   15A3    .    .    .   ;*******************************************
5973   15A3    .    .    .   ; IORMG1 - CHECK FOR PRESENCE OF OPTION ROM *
5974   15A3    .    .    .   ;*******************************************
5975   15A3    .    .    .   ;
5976   15A3    .    .    .   ;  ENTRY:  H,L = ROM STARTING ADDRESS
5977   15A3    .    .    .   ;
5978   15A3    .    .    .   ;  EXIT :  Z => ROM EXIST
5979   15A3    .    .    .   ;            H,L = H,L(ENTRY)+1
5980   15A3    .    .    .   ;          NZ => ROM ABSENT
5981   15A3    .    .    .   ;          A DESTROYED
5982   15A3    .    .    .   ;            H,L = H,L(ENTRY) => ROM ABSENT
5983   15A3    .    .    .   ;            H,L = H,L(ENTRY)+1 => WRONG ROM
5984   15A3    .    .    .   ;
5985   15A3    .    .    .   IORMG1 EQU  $
5986   15A3    7E   .    .          MOV  A,M        ;GET FIRST ROM BYTE
5987   15A4    E6   F0   .          ANI  360Q       ;CHECK UPPER 4 BITS ONLY
5988   15A6    FE   50   .          CPI  P          ;IS IT AN UPPER CASE P?
5989   15A8    C0   .    .          RNZ             ;NO - RETURN ROM ABSENT
5990   15A9    23   .    .          INX  H          ;YES - CHECK SECOND BYTE
5991   15AA    7E   .    .          MOV  A,M        ;SECOND BYTE OF ROM SHOULD
5992   15AB    BC   .    .          CMP  H            ;EQUAL HIGH ORDER EIGHT
5993   15AC    C9   .    .          RET               ;BITS IN ITS PROPER ADDRES
5994   15AD    .    .    .   ;                         RANGE
```

```
 5996    15AD    .    .    .   ;************************************
 5997    15AD    .    .    .   ; IOINTR - I/O INTERRUPT PROCESSING *
 5998    15AD    .    .    .   ;************************************
 5999    15AD    .    .    .   ;
 6000    15AD    .    .    .   ;  ENTRY:   "PSW" AND H,L PUSHED
 6001    15AD    .    .    .   ;           A = INTERRUPT CODE
 6002    15AD    .    .    .   ;
 6003    15AD    .    .    .   IOINTR EQU  $
 6004    15AD    CD   65   91        CALL INTVEC     ;CHECK ALTERNATE INTERRUPT
 6005    15B0    3A   F5   FF        LDA  PRCCTL     ;GET CURRENT PROCESSOR STATE
 6006    15B3    F6   40   .         ORI  POLL       ;POLL THE I/O BOARDS TO FIND
 6007    15B5    D3   70   .         OUT  PROCSR       ;OUT WHO INTERRUPTED
 6008    15B7    21   00   87        LXI  H,IOCRCL   ;DO DUMMY I/O READ TO GET
 6009    15BA    6E   .    .         MOV  L,M          ;POLL RESPONSE
 6010    15BB    E6   BF   .         ANI  377Q-POLL
 6011    15BD    D3   70   .         OUT  PROCSR     ;RESTORE PROCESSOR STATE
 6012    15BF    3A   7F   FE        LDA  DEVFLG     ;GET POLL DEVICE FLAG
 6013    15C2    A5   .    .         ANA  L          ;DEVICE DRIVER PRESENT?
 6014    15C3    FA   3D   28        JM   CTINTR     ;CTU - DO CTU ROUTINE
 6015    15C6    87   .    .         ADD  A          ;ALTERNATE I/O INTERRUPT?
 6016    15C7    FA   08   60        JM   ZINTAL     ;YES - GO CHECK INTERRUPT
 6017    15CA    .    .    .   ;******************************************
 6018    15CA    .    .    .   ; INVALID DEVICE INTERRUPT - REPORT ERROR *
 6019    15CA    .    .    .   ;******************************************
 6020    15CA    7D   .    .         MOV  A,L        ;RECALL POLL RESPONSE
 6021    15CB    06   40   .         MVI  B,ATSIGN   ;COMPUTE ERROR CODE
 6022    15CD    B7   .    .         ORA  A          ;ANY DEVICE INTERRUPTED?
 6023    15CE    CA   D6   15        JZ   IOI020     ;NO - DON'T LOOK FOR BIT
 6024    15D1    .    .    .   ;                      YES - DETERMINE DEVICE
 6025    15D1    .    .    .   IOI010 EQU  $
 6026    15D1    04   .    .         INR  B          ;INCREMENT ERROR CODE
 6027    15D2    07   .    .         RLC             ;DEVICE TYPE FOUND?
 6028    15D3    D2   D1   15        JNC  IOI010     ;NO - CONTINUE LOOKING
 6029    15D6    .    .    .   IOI020 EQU  $          ;YES - SET ERROR CODE
 6030    15D6    78   .    .         MOV  A,B
 6031    15D7    .    .    .   ;                      FALL INTO ERROR REPORTER
```

```
=========================================================================
ITEM   LOC   OBJECT CODE  SOURCE STATEMENTS
=========================================================================
6033   15D7   .   .   .   ;*******************************************
6034   15D7   .   .   .   ; INTERR - REPORT INVALID INTERRUPT OCCURRED *
6035   15D7   .   .   .   ;*******************************************
6036   15D7   .   .   .   ;
6037   15D7   .   .   .   ;  ENTRY:  A = ERROR CODE (ASCII CHARACTER)
6038   15D7   .   .   .   ;
6039   15D7   .   .   .   INTERR EQU  $
6040   15D7   21  DE  FF         LXI  H,IODATA  ;SET ERROR CODE FOR ERROR
6041   15DA   22  EF  FF         SHLD MSGPT2       ;MESSAGE
6042   15DD   77  .   .          MOV  M,A
6043   15DE   23  .   .          INX  H
6044   15DF   36  CE  .          MVI  M,EOP
6045   15E1   21  3F  0F         LXI  H,INERMS  ;REPORT INTERRUPT ERROR
6046   15E4   AF  .   .          XRA  A         ;STOP ANY CTU MOTION
6047   15E5   32  00  8B         STA  IOCTCO
6048   15E8   C3  54  12         JMP  HANGUO       ;AND HANG TERMINAL
6049   15EB   .   .   .   ;****************************************
6050   15EB   .   .   .   ; INTRPT - PROCESS UNEXPECTED INTERRUPTS *
6051   15EB   .   .   .   ;****************************************
6052   15EB   .   .   .   ;
6053   15EB   .   .   .   ;  ENTRY:  "PSW" PUSHED
6054   15EB   .   .   .   ;          A = INTERRUPT CODE
6055   15EB   .   .   .   ;          C-FLAG CLEARED
6056   15EB   .   .   .   ;
6057   15EB   .   .   .   INTRPT EQU  $
6058   15EB   CD  65  91         CALL INTVEC    ;ANY INTERRUPT HANDLER?
6059   15EE   D2  D7  15         JNC  INTERR    ;NO - REPORT ERROR
6060   15F1   F1  .   .          POP  PSW       ;YES - RESTORE PSW
6061   15F2   FB  .   .          EI             ;RE-ENABLE INTERRUPTS
6062   15F3   C9  .   .          RET            ;RETURN TO INTERRUPTED CODE
```

```
 6064   15F4    .    .    .   ;**************************************
 6065   15F4    .    .    .   ; IOINTR - I/O INTERRUPT PROCESSING *
 6066   15F4    .    .    .   ;**************************************
 6067   15F4    .    .    .   ;******************
 6068   15F4    .    .    .   ; KEYBOARD ENABLE *
 6069   15F4    .    .    .   ;******************
 6070   15F4    .    .    .   KBEN    EQU $
 6071   15F4    3A   6E   FF          LDA  DFLGS
 6072   15F7    E6   40   .           ANI KBDLOK     ;KEYBOARD LOCKED BY ESC SEQ?
 6073   15F9    C0   .    .           RNZ            ;YES - DO NOT UNLOCK KEYBOARD
 6074   15FA    .    .    .   ;
 6075   15FA    .    .    .   KBEN1   EQU $
 6076   15FA    3E   02   .           MVI  A,UNLKKB  ;UNLOCK THE KEYBOARD
 6077   15FC    CD   08   48          CALL ZKBCTL
 6078   15FF    3E   BF   .           MVI  A,377Q-KBDLOK  ;CLEAR LOCKED FLAG
 6079   1601    .    .    .   ;
 6080   1601    .    .    .   ;  CLRDFL - CLEAR DATA TRANSFER FLAGS
 6081   1601    .    .    .   ;
 6082   1601    .    .    .   ;       ENTRY:  A = FLAGS TO BE CLEARED
 6083   1601    .    .    .   ;
 6084   1601    .    .    .   CLRDFL EQU  $
 6085   1601    21   6E   FF          LXI  H,DFLGS
 6086   1604    A6   .    .           ANA  M             ;MASK OUT FLAGS
 6087   1605    .    .    .   ;
 6088   1605    .    .    .   ;  STOREA - STORE VALUE N A-REG AND RETURN
 6089   1605    .    .    .   ;
 6090   1605    .    .    .   ;       ENTRY:  A = VALUE TO BE STORED
 6091   1605    .    .    .   ;             H,L = LOCATION TO BE STORED IN
 6092   1605    .    .    .   ;
 6093   1605    .    .    .   STOREA EQU $
 6094   1605    77   .    .          MOV M,A        ;STORE UPDATED VALUE
 6095   1606    C9   .    .          RET            ;RETURN
 6096   1607    .    .    .   ;****************
 6097   1607    .    .    .   ; KEYBOARD LOCK *
 6098   1607    .    .    .   ;****************
 6099   1607    .    .    .   KBLOKO EQU $
 6100   1607    3E   40   .           MVI  A,KBDLOK  ;SET ESCAPE SEQUENCE LOCK
 6101   1609    CD   11   17          CALL SETDFL       ;FLAG
 6102   160C    .    .    .   ;
 6103   160C    .    .    .   KBLOK   EQU $
 6104   160C    3E   01   .           MVI  A,LOCKKB  ;LOCK THE KEYBOARD
 6105   160E    C3   08   48          JMP  ZKBCTL
```

```
6107   1611   .   .   .   ;*****************************
6108   1611   .   .   .   ;         ESC & LOWER CASE B     *
6109   1611   .   .   .   ;        BINARY LOADER           *
6110   1611   .   .   .   ;  A    SET ADDRESS = DIGITS     *
6111   1611   .   .   .   ;  C    COMPARE CHECKSUM         *
6112   1611   .   .   .   ;  D     STORE BYTE              *
6113   1611   .   .   .   ;        INCREMENT ADDRESS       *
6114   1611   .   .   .   ;  E    CALL ADDRESS             *
6115   1611   .   .   .   ;  DIGITS   1,2,3,4, OR 5        *
6116   1611   .   .   .   ;*****************************
6117   1611   .   .   .   LOADR  EQU  $           ;INITIAL ENTRY
6118   1611   3E  18  .          MVI  A,MAXROW+1
6119   1613   32  C0  FF         STA  CURROW      ;SET CURSOR OFF THE SCREEN
6120   1616   21  4A  0F         LXI  H,LDRMSG
6121   1619   CD  D6  1C         CALL DSPMS0      ;DISPLAY THE LOADER MESSAGE
6122   161C   .   .   .   LOADR1 EQU  $           ;ENTRY TO NOT DISPLAY MESSAG
6123   161C   21  00  00         LXI  H,0         ;CLEAR CHECKSUM ACCUMULATOR
6124   161F   22  D7  FF         SHLD LCHKSM
6125   1622   3E  04  .          MVI  A,FRCRST    ;SET FORCE RESET FLAG
6126   1624   CD  00  14         CALL STCMFL
6127   1627   .   .   .   LDR0   EQU $
6128   1627   3A  88  FF         LDA  CHAR        ;RECALL INPUT CHARACTER
6129   162A   E6  20  .          ANI  40Q         ;IS IT UPPER CASE?
6130   162C   3E  FB  .          MVI  A,377Q-FRCRST
6131   162E   CA  DC  13         JZ   CLCMFL      ;YES - CLEAR FORCE RESET AND
6132   1631   .   .   .   ;                          EXIT ESCAPE SEQUENCE
6133   1631   21  9C  27         LXI  H,LDRTAB    ;NO - SET LOADER FUNCTION
6134   1634   3E  08  .          MVI  A,OCTRDX    ;SET FOR OCTAL RADIX
6135   1636   C3  81  04         JMP  ESCAP0
6136   1639   .   .   .   ;
6137   1639   .   .   .   ;  <A> - ADDRESS PARAMETER - SET ADDRESS
6138   1639   .   .   .   ;
6139   1639   .   .   .   LDR3   EQU $
6140   1639   2A  DE  FF         LHLD LDATA   ;GET ACCUMULATED DATA
6141   163C   22  D5  FF         SHLD LADDR   ;SET AS LOAD ADDRESS
6142   163F   EB  .   .          XCHG             ;PUT VALUE INTO D,E
6143   1640   .   .   .   LDR035 EQU $
6144   1640   2A  D7  FF         LHLD LCHKSM      ;ACCUMULATE CHECKSUM
6145   1643   19  .   .          DAD  D
6146   1644   22  D7  FF         SHLD LCHKSM
6147   1647   C3  27  16         JMP LDR0       ;RETURN TO SYSTEM
```

```
==========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
==========================================================================
 6149   164A    .    .    .   ;
 6150   164A    .    .    .   ;  <D> - DATA BYTE PARAMETER - STORE DATA BYTE
 6151   164A    .    .    .   ;
 6152   164A    .    .    .   LDR4    EQU $
 6153   164A   2E   DE    .           MVI  L,LDATA-BASE
 6154   164C   5E    .    .           MOV  E,M         ;GET ACCUMULATED DATA
 6155   164D   2A   D5   FF           LHLD LADDR       ;GET LOAD ADDRESS
 6156   1650   73    .    .           MOV M,E        ;STORE THE BYTE
 6157   1651   16   00    .           MVI D,0        ;ZERO MSB FOR CHECKSUM
 6158   1653   23    .    .           INX  H           ;INCREMENT AND STORE NEW
 6159   1654   22   D5   FF           SHLD LADDR         ;LOAD ADDRESS
 6160   1657   C3   40   16           JMP  LDR035      ;ACCUMULATE CHECKSUM
 6161   165A    .    .    .   ;*********************************************
 6162   165A    .    .    .   ; <E> - EXECUTE ENTERED CODE, WAIT UNTIL CTU'S *
 6163   165A    .    .    .   ;   STOPPED BEFORE EXECUTING CODE            *
 6164   165A    .    .    .   ;*********************************************
 6165   165A    .    .    .   LDR060 EQU   $
 6166   165A   CD   A5   0F          CALL DISLN4      ;RE-ENABLE RESET KEY
 6167   165D   CD   17   50          CALL ZGETDC      ;PURGE DATA COMM INPUT
 6168   1660   DC   51   12          CC   DCERR       ;PROCESS ERROR IF ANY
 6169   1663   3A   55   FF          LDA  CMND        ;GET CTU COMMAND
 6170   1666   E6   01    .          ANI  RUN         ;CTU'S RUNNING?
 6171   1668   C2   5A   16          JNZ  LDR060      ;YES - CONTINUE WAITING
 6172   166B   3E   80    .          MVI  A,CRTOFF    ;NO - TURN OFF THE DISPLAY
 6173   166D   32   20   87          STA  IOCRRW
 6174   1670   F3    .    .          DI               ;DISABLE INTERRUPTS
 6175   1671   2A   D5   FF          LHLD LADDR       ;GET LOAD ADDRESS
 6176   1674   E9    .    .          PCHL             ;START EXECUTION THERE
 6177   1675    .    .    .   ;
 6178   1675    .    .    .   ;  <C> - CHECKSUM ENTRY
 6179   1675    .    .    .   ;
 6180   1675    .    .    .   LDR10   EQU $         ;CHECKSUM ENTRY
 6181   1675   21   F7   FF          LXI  H,ERRFLG  ;DEFAULT TO GOOD CHECKSUM
 6182   1678   7E    .    .          MOV  A,M
 6183   1679   F6   04    .          ORI  LDRCHK
 6184   167B   77    .    .          MOV  M,A         ;SET ERROR FLAGS
 6185   167C   2A   DE   FF          LHLD LDATA       ;GET USER SPECIFIED CHECKSUM
 6186   167F   EB    .    .          XCHG
 6187   1680   21   D7   FF          LXI  H,LCHKSM
 6188   1683   7B    .    .          MOV  A,E         ;COMPARE TO CALCULATED
 6189   1684   AE    .    .          XRA  M              ;CHECKSUM
 6190   1685   4F    .    .          MOV  C,A
 6191   1686   23    .    .          INX  H
 6192   1687   7A    .    .          MOV  A,D
 6193   1688   AE    .    .          XRA  M
 6194   1689   B1    .    .          ORA  C           ;DO CHECKSUMS MATCH?
 6195   168A   CA   27   16          JZ   LDR0        ;YES - RETURN NORMAL
 6196   168D   C7    .    .          RST  ;RESET       NO - RESET TERMINAL
```

```
================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
================================================================================
 6198    168E   .    .    .   ;*****************************************
 6199    168E   .    .    .   ; PARAMETERIZED SEQUENCES INITIAL CONTROL *
 6200    168E   .    .    .   ;*****************************************
 6201    168E   .    .    .   PRMSEQ EQU  $
 6202    168E   21   32   27         LXI  H,PRMTAB  ;SET RANGE TABLE FOR
 6203    1691   C3   7F   04         JMP  ESCAPA      ;PARAMETERIZED ESC SEQUENC
 6205   1694   .   .   .   ;****************
 6206   1694   .   .   .   ; START PROTECT *
 6207   1694   .   .   .   ;****************
 6208   1694   .   .   .   PRSTRT EQU $
 6209   1694   06  C0  .          MVI  B,STPR     ;STORE START PROTECT CONTROL
 6210   1696   C3  B9  16         JMP  PRO100       ;FLAG
 6211   1699   .   .   .   ;****************
 6212   1699   .   .   .   ; TRANSMIT-ONLY *
 6213   1699   .   .   .   ;****************
 6214   1699   .   .   .   STRXMO EQU  $
 6215   1699   3E  C2  .          MVI  A,XMONLY  ;STORE TRANSMIT-ONLY CONTROL
 6216   169B   C3  A0  16         JMP  PRO010       ;FLAG
 6217   169E   .   .   .   ;**************
 6218   169E   .   .   .   ; END PROTECT *
 6219   169E   .   .   .   ;**************
 6220   169E   .   .   .   PREND  EQU $
 6221   169E   3E  C1  .          MVI  A,ENDPR    ;STORE END PROTECT CONTROL
 6222   16A0   .   .   .   ;
 6223   16A0   .   .   .   ;  MAKE SURE PREVIOUS CHAR IS DEFINED PROTECTED
 6224   16A0   .   .   .   ;
 6225   16A0   .   .   .   PRO010 EQU  $
 6226   16A0   32  DB  FF         STA  PARM1      ;SAVE CONTROL FLAG
 6227   16A3   3A  C1  FF         LDA  CURCOL     ;GET THE CURRENT COLUMN
 6228   16A6   3D  .   .          DCR  A          ;SET TO FIND PREVIOUS COLUMN
 6229   16A7   CD  0B  07         CALL RCADRO     ;PREVIOUS COLUMN PRESENT?
 6230   16AA   FA  14  48         JM   ZBELL      ;NO - SOUND BELL AND RETURN
 6231   16AD   3A  C5  FF         LDA  LSTFMT     ;YES - RECALL LAST FORMAT CT
 6232   16B0   FE  C0  .          CPI  STPR       ;WAS IT A START PROTECT?
 6233   16B2   C4  94  16         CNZ  PRSTRT     ;NO - ENTER STPR
 6234   16B5   3A  DB  FF         LDA  PARM1      ;RECALL FORMAT CONTROL FLAG
 6235   16B8   47  .   .          MOV  B,A          ;TO BE STORED
 6236   16B9   .   .   .   ;
 6237   16B9   .   .   .   ;  ENTER THE FORMAT CONTROL FLAG
 6238   16B9   .   .   .   ;
 6239   16B9   .   .   .   PRO100 EQU $
 6240   16B9   CD  76  19         CALL CHKFMS   ;FORMAT MODE?
 6241   16BC   C0  .   .          RNZ           ;YES - TERMINATE
 6242   16BD   78  .   .          MOV  A,B        ;NO - ADD CHAR TO DISPLAY
 6243   16BE   F5  .   .          PUSH PSW        ;SAVE THE CONTROL CODE
 6244   16BF   CD  E0  21         CALL DISPC1       ;(DISPC1 DESTROYS "LSTFMT"
 6245   16C2   F1  .   .          POP  PSW        ;RECALL CONTROL CODE
 6246   16C3   32  C5  FF         STA  LSTFMT       ;NEW ENTRY
 6247   16C6   C9  .   .          RET
```

```
==========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                    PAGE 181
==========================================================================
 6249   16C7   .    .    .   ;
 6250   16C7   .    .    .   ;  ENTREN - ENABLE ENTER VIA ESCAPE SEQUENCE
 6251   16C7   .    .    .   ;
 6252   16C7   .    .    .   ENTREN EQU  $
 6253   16C7   01   00   40        LXI  B,SENTER  ;SET DISPLAY SEND PENDING
 6254   16CA   .    .    .   ;                     FALL INTO "SBLXFO"
 6255   16CA   .    .    .   ;*********************************************
 6256   16CA   .    .    .   ; SBLXFO - SET BLOCK TRANSFER FLAG FOR ESCAPE *
 6257   16CA   .    .    .   ;   SEQUENCE INITIATED BLOCK TRANSFERS        *
 6258   16CA   .    .    .   ;*********************************************
 6259   16CA   .    .    .   ;
 6260   16CA   .    .    .   ;  ENTRY:  B = FLAG TO BE SET IN "MFLGS"
 6261   16CA   .    .    .   ;          C = FLAG TO BE SET IN "MFLGS2"
 6262   16CA   .    .    .   ;
 6263   16CA   .    .    .   ;  EXIT :  ALL REGISTERS DESTROYED
 6264   16CA   .    .    .   ;          X-ON AND DC2 PENDING FLAGS ARE SET
 6265   16CA   .    .    .   ;          ACCORDING TO THE SETTINGS OF G AND H
 6266   16CA   .    .    .   ;
 6267   16CA   .    .    .   SBLXFO EQU  $
 6268   16CA   CD   88   10        CALL CLRXON    ;CLEAR BLOCK TRANSFER TRIGGE
 6269   16CD   .    .    .   ;
 6270   16CD   .    .    .   ;  SBLXFA - DETERMINE DC2 HANDSHAKE MODE FOR
 6271   16CD   .    .    .   ;     NON-BLOCK MODE KEYBOARD INITIATED BLOCK
 6272   16CD   .    .    .   ;     TRANSFERS
 6273   16CD   .    .    .   ;
 6274   16CD   .    .    .   SBLXFA EQU $
 6275   16CD   3A   FB   FF        LDA  KBJMPR    ;GET THE STRAP SETTINGS
 6276   16D0   E6   40   .         ANI HNDSHK   ;DC2 ON ALL BLOCK TRANSFERS?
 6277   16D2   CA   E2   16        JZ  SBL010   ;NO - DO NOT SET DC2 FLAG
 6278   16D5   .    .    .   ;                 YES - FALL INTO "SBLXF1"
```

```
===========================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS
===========================================================================
 6280   16D5    .    .    .   ;**********************************************
 6281   16D5    .    .    .   ; SBLXF1 - SET BLOCK TRANSFER FLAG FOR KEYBOARD *
 6282   16D5    .    .    .   ;   INITIATED BLOCK TRANSFERS                   *
 6283   16D5    .    .    .   ;**********************************************
 6284   16D5    .    .    .   ;
 6285   16D5    .    .    .   ;  ENTRY:  B = FLAG TO BE SET IN "MFLGS"
 6286   16D5    .    .    .   ;          C = FLAG TO BE SET IN "MFLGS2"
 6287   16D5    .    .    .   ;
 6288   16D5    .    .    .   SBLXF1 EQU   $
 6289   16D5    3A   FB   FF         LDA   KBJMPR     ;GET THE STRAP SETTINGS
 6290   16D8    E6   80   .          ANI   DC2SND     ;INHIBIT DC2 HANDSHAKE?
 6291   16DA    3E   01   .          MVI   A,SDC2/256  ;(SET DC2 PENDING FLAG)
 6292   16DC    CA   E3   16         JZ    SBL020     ;NO - SET DC2 PENDING FLAG
 6293   16DF    CD   24   04         CALL  CHKCT1     ;YES - SET BLOCK TRANSFER
 6294   16E2    .    .    .   ;                          TRIGGER TO CAUSE IMMEDIATE
 6295   16E2    .    .    .   ;                          TRANSMISSION OF DATA
 6296   16E2    .    .    .   SBL010 EQU $
 6297   16E2    78   .    .          MOV A,B        ;PUT FLAG INTO A-REGISTER
 6298   16E3    .    .    .   SBL020 EQU   $
 6299   16E3    B0   .    .          ORA   B          ;ADD IN OPTIONAL DC2 FLAG
 6300   16E4    47   .    .          MOV   B,A        ;SAVE FLAGS IN B-REGISTER
 6301   16E5    CD   6A   10         CALL  CKRMTE     ;REMOTE MODE ENABLED?
 6302   16E8    C8   .    .          RZ               ;NO - DON'T SET BLOCK XFR
 6303   16E9    21   70   FF         LXI   H,MFLGS    ;YES - SET DATA PENDING
 6304   16EC    78   .    .          MOV   A,B          ;FLAGS
 6305   16ED    B6   .    .          ORA   M
 6306   16EE    77   .    .          MOV   M,A
 6307   16EF    2B   .    .          DCX   H
 6308   16F0    79   .    .          MOV   A,C
 6309   16F1    B6   .    .          ORA   M          ;SET FLAG IN "MFLGS2"
 6310   16F2    77   .    .          MOV   M,A
 6311   16F3    C3   0C   16         JMP   KBLOK      ;DISABLE THE KEYBOARD
 6313    16F6    .    .    .   ;*************************************
 6314    16F6    .    .    .   ; SDTRM1 - SEND TERMINATOR CHARACTER *
 6315    16F6    .    .    .   ;*************************************
 6316    16F6    .    .    .   ;
 6317    16F6    .    .    .   ;  EXIT :  A DESTROYED
 6318    16F6    .    .    .   ;
 6319    16F6    .    .    .   SDTRM1 EQU   $
 6320    16F6    CD   59   10         CALL  GTMODE     ;PAGE MODE?
 6321    16F9    3A   04   50         LDA   BLKTRM       ;(GET BLOCK TERMINATOR)
 6322    16FC    C2   C1   17         JNZ   XPUTDC     ;YES - SEND BLOCK TERM ONLY
 6323    16FF    .    .    .   SDTRM2 EQU   $          ;NO - SEND CR(LF)
 6324    16FF    3E   0D   .          MVI   A,CR
 6325    1701    CD   C1   17         CALL  XPUTDC     ;TRANSMIT RETURN
 6326    1704    3A   F3   FF         LDA   MDFLG2
 6327    1707    E6   04   .          ANI   AUTOLF     ;AUTO LINE FEED ENABLED?
 6328    1709    C8   .    .          RZ               ;NO - RETURN
 6329    170A    .    .    .   SDTRM3 EQU   $
 6330    170A    3E   0A   .          MVI   A,LF       ;YES - TRANSMIT LINE FEED
 6331    170C    C3   C1   17         JMP   XPUTDC
 6333    170F     .    .    .    ;*********************************
 6334    170F     .    .    .    ; SETDFL - SET DATA TRANSFER FLAG *
 6335    170F     .    .    .    ;*********************************
 6336    170F     .    .    .    ;
 6337    170F     .    .    .    ;  ENTRY:  A = FLAG BIT TO BE SET
 6338    170F     .    .    .    ;
 6339    170F     .    .    .    ;  EXIT :  H = BASEH
 6340    170F     .    .    .    ;          A,L DESTROYED
 6341    170F     .    .    .    ;
 6342    170F     .    .    .    SETDFO EQU  $          ;SET DATA COMM INPUT FLAG
 6343    170F     3E   01   .           MVI  A,SDACOM   ;SET FLAG BIT TO BE SET
 6344    1711     .    .    .    SETDFL EQU  $
 6345    1711     21   6E   FF          LXI  H,DFLGS
 6346    1714     B6   .    .           ORA  M          ;MERGE FLAG BIT TO EXISTING
 6347    1715     77   .    .           MOV  M,A          ;FLAGS
 6348    1716     C9   .    .           RET             ;RETURN
 6350   1717    .    .    .   ;
 6351   1717    .    .    .   ; * * * * * * * * * * * * * * * * * * * * * * * *
 6352   1717    .    .    .   ;
 6353   1717    .    .    .   ;  SETLFT,SETRHT - SET LEFT AND RIGHT MARGINS
 6354   1717    .    .    .   ;
 6355   1717    .    .    .   ;       ENTRY:  H = BASEH
 6356   1717    .    .    .   ;               CURCOL = CURSOR COLUMN POSITION
 6357   1717    .    .    .   ;
 6358   1717    .    .    .   ;       EXIT :  LFTMGN,RHTMGN SET APPROPRIATELY
 6359   1717    .    .    .   ;
 6360   1717    .    .    .   SETLFT EQU  $
 6361   1717   CD   7B   19          CALL CHKFMT     ;FORMAT MODE?
 6362   171A   C0    .    .          RNZ             ;YES - DON'T SET MARGIN
 6363   171B   3A   BE   FF          LDA  RHTMGN     ;NO - GET THE RIGHT MARGIN
 6364   171E   2E   C1    .          MVI  L,CURCOL-BASE
 6365   1720   BE    .    .          CMP  M          ;CURSOR AFTER RIGHT MARGIN?
 6366   1721   FA   51   23          JM   DSPCH1     ;YES - DON'T SET MARGIN
 6367   1724   7E    .    .          MOV  A,M        ;NO - SET NEW LEFT MARGIN
 6368   1725   32   BF   FF          STA  LFTMGN
 6369   1728   C9    .    .          RET             ;RETURN
 6370   1729    .    .    .   ;
 6371   1729    .    .    .   SETRHT EQU  $
 6372   1729   CD   7B   19          CALL CHKFMT     ;FORMAT MODE?
 6373   172C   C0    .    .          RNZ             ;YES - DON' SET MARGIN
 6374   172D   3A   C1   FF          LDA  CURCOL     ;GET CURRENT CURSOR COLUMN
 6375   1730   2E   BF    .          MVI  L,LFTMGN-BASE
 6376   1732   BE    .    .          CMP  M          ;BEFORE LEFT MARGIN?
 6377   1733   FA   51   23          JM   DSPCH1     ;YES - DON'T SET MARGIN
 6378   1736   2B    .    .          DCX  H          ;NO - SET NEW RIGHT MARGIN
 6379   1737   77    .    .          MOV  M,A
 6380   1738   C9    .    .          RET             ;RETURN
```

```
===========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
===========================================================================
 6333   170F    .   .   .   ;**********************************
 6334   170F    .   .   .   ; SETDFL - SET DATA TRANSFER FLAG *
 6335   170F    .   .   .   ;**********************************
 6336   170F    .   .   .   ;
 6337   170F    .   .   .   ;  ENTRY:  A = FLAG BIT TO BE SET
 6338   170F    .   .   .   ;
 6339   170F    .   .   .   ;  EXIT :  H = BASEH
 6340   170F    .   .   .   ;          A,L DESTROYED
 6341   170F    .   .   .   ;
 6342   170F    .   .   .   SETDFO EQU  $          ;SET DATA COMM INPUT FLAG
 6343   170F    3E  01  .          MVI  A,SDACOM   ;SET FLAG BIT TO BE SET
 6344   1711    .   .   .   SETDFL EQU  $
 6345   1711    21  6E  FF         LXI  H,DFLGS
 6346   1714    B6  .   .          ORA  M          ;MERGE FLAG BIT TO EXISTING
 6347   1715    77  .   .          MOV  M,A          ;FLAGS
 6348   1716    C9  .   .          RET             ;RETURN
```

```
6397    173F    .   .   .   ;*****************************************
6398    173F    .   .   .   ; SETTRM - SET NON-DISPLAYING TERMINATOR *
6399    173F    .   .   .   ;*****************************************
6400    173F    .   .   .   SETTRM EQU  $
6401    173F   3E  01   .          MVI  A,IGNTRM  ;SET TO IGNORE NON-DISPLAYIN
6402    1741   32  6D  FF          STA  TRMFCT      ;TERMINATORS
6403    1744   3E  C4   .          MVI  A,STPFLG  ;ADD NON-DISPLAYING
6404    1746   CD  E2  21          CALL DISPC2      ;TERMINATOR TO DISPLAY
6405    1749   C3  20  1E          JMP  FLDSRX    ;SET "LSTCOL" TO MAXCOL+1
6406    174C    .   .   .   ;                       TO FORCE LINE RE-SCAN TO
6407    174C    .   .   .   ;                       INHIBIT DELETION OF NEW
6408    174C    .   .   .   ;                       NON-DISPLAYING TERMINATOR
```

```
===========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
===========================================================================
 6410   174C    .   .   .   ;*****************************************
 6411   174C    .   .   .   ; SNDCDE - SEND ATTENTION/FUNCTION CODE *
 6412   174C    .   .   .   ;*****************************************
 6413   174C    .   .   .   SNDCDE EQU  $
 6414   174C   3A  6E  FF          LDA  DFLGS      ;GET DATA TRANSFER FLAGS
 6415   174F   E6  01   .          ANI  SDACOM     ;COMMAND FROM DATA COMM?
 6416   1751   C0   .   .          RNZ             ;YES - IGNORE IT
 6417   1752   21  80  27          LXI  H,SNDCTB   ;SET TO ACCUMULATE OCTAL
 6418   1755   3E  08   .          MVI  A,OCTRDX     ;CODE CHARACTER
 6419   1757   C3  81  04          JMP  ESCAP0
 6420   175A    .   .   .   ;****************************
 6421   175A    .   .   .   ; <A> - SEND ATTENTION CODE *
 6422   175A    .   .   .   ;****************************
 6423   175A    .   .   .   SNDCD1 EQU  $
 6424   175A   3A  DE  FF          LDA  IODATA     ;GET ACCUMULATED VALUE
 6425   175D   47   .   .          MOV  B,A        ;PUT CODE INTO B-REGISTER
 6426   175E   3E  0B   .          MVI  A,SNDATN   ;SET DATA COMM CONTROL CODE
 6427   1760   C3  42  12          JMP  DCMCTL     ;PERFORM FUNCTION
```

```
1763  .    .    .   ;*********************************************
1763  .    .    .   ; STRTBL - SET FIRST DISPLAY OUT CHARACTER FOR *
1763  .    .    .   ;   BLOCK STORE                               *
1763  .    .    .   ;*********************************************
1763  .    .    .   ;
1763  .    .    .   ;  ENTRY:  DON'T CARE
1763  .    .    .   ;
1763  .    .    .   ;  EXIT :  CURCOL,CURROW = STARTING POSITION
1763  .    .    .   ;
1763  .    .    .   ;  IF THE AUTO TERMINATOR STRAP (J) IS OUT, A
1763  .    .    .   ;  TERMINATOR IS PLACED AHEAD OF THE CURRENT
1763  .    .    .   ;  CURSOR POSITION AND A REVERSE SCAN IS MADE
1763  .    .    .   ;  FOR THE FIRST TERMINATOR BEFORE THE CURRENT
1763  .    .    .   ;  CURSOR POSITION.  OTHERWISE, THE CURSOR IS
1763  .    .    .   ;  PLACED AT THE HOME POSITION
1763  .    .    .   ;
1763  .    .    .   STRTBL EQU  $
1763  CD   69   17         CALL STRTB1    ;SET CURSOR START POSITION
1766  C3   9C   25         JMP  INITDG    ;SET UP DISPLAY GET ROUTINE
1769  .    .    .   ;
1769  .    .    .   STRTB1 EQU  $
1769  3A   FA   FF         LDA  KBJMP2    ;GET KEYBOARD JUMPERS 2
176C  E6   01   .          ANI  AUTTRM    ;AUTO TERMINATOR ENABLED?
176E  CA   F4   17         JZ   XMOHME    ;NO - HOME THE CURSOR
1771  .    .    .   ;*******************************
1771  .    .    .   ; STTERM - SET AUTO TERMINATOR *
1771  .    .    .   ;*******************************
1771  .    .    .   ;
1771  .    .    .   ;  EXIT :  Z => AUTO TERMINATOR NOT SET
1771  .    .    .   ;         NZ => AUTO TERMINATOR SET
1771  .    .    .   ;
1771  .    .    .   STTERM EQU  $
1771  CD   81   19         CALL CHKMLK    ;MEMORY LOCK ENABLED?
1774  CA   7D   17         JZ   STB010    ;YES - CHECK FOR FREE BLOCKS
1777  CD   7B   19         CALL CHKFMT    ;FORMAT MODE ENABLED?
177A  CA   84   17         JZ   STB050    ;NO - ADD TERMINATOR
177D  .    .    .   STB010 EQU  $         ;YES - CHECK FOR FREE BLOCKS
177D  3A   AC   FF         LDA  FRBLKS    ;GET LSB OF FREE BLOCKS PTR
1780  B7   .    .          ORA  A         ;ANY FREE BLOCKS?
1781  CA   07   0B         JZ   MLOCK     ;NO - FORCE MEMORY LOCK ON
```

```
1784   .  .  .   ;*****************************************
1784   .  .  .   ; SPACE AVAILABLE - STORE NON-DISPLAYING  *
1784   .  .  .   ;   TERMINATOR AT CURRENT CURSOR POSITION *
1784   .  .  .   ;*****************************************
1784   .  .  .   STB050 EQU  $
1784   CD 3F 17         CALL SETTRM     ;STORE TERMINATOR
1787   3A 04 50         LDA  BLKTRM     ;GET BLOCK TERMINATOR CHAR
178A   6F .  .          MOV  L,A        ;SET PARAMETERS FOR REVERSE
178B   26 C4 .          MVI  H,STPFLG     ;SEARCH FOR PREV TERMINATO
178D   CD 05 18         CALL BACKT1     ;IS THER A PREV TERMINATOR?
1790   C2 A2 17         JNZ  STB080     ;NO - HOME THE CURSOR
1793   CD A4 06         CALL RCADRA     ;DOES THE CHARACTER EXIST?
1796   C4 96 20         CNZ  CRLF       ;NO - START AT NEXT LINE
1799   .  .  .   STB060 EQU  $
1799   3E FB .          MVI  A,377Q-NOSEND
179B   CD 01 16         CALL CLRDFL     ;CLEAR NO DATA FLAG
179E   F6 08 .          ORI  SKPTRM     ;SET TO SKIP INITIAL BLOCK
17A0   77 .  .          MOV  M,A          ;TERMINATOR CHARACTER
17A1   C9 .  .          RET             ;RETURN NZ
```

```
=======================================================================
ITEM     LOC   OBJECT CODE  SOURCE STATEMENTS                    PAGE 191
=======================================================================
6490    17A2    .    .    .   ;****************************************
6491    17A2    .    .    .   ; NO PREVIOUS TERMINATOR - HOME THE CURSOR *
6492    17A2    .    .    .   ;****************************************
6493    17A2    .    .    .   STB080 EQU  $
6494    17A2    2A   C0   FF         LHLD CURROW    ;SAVE THE CURRENT ROW AND
6495    17A5    E5   .    .          PUSH H           ;COLUMN VALUES
6496    17A6    CD   31   13         CALL DPSEN1    ;HOME CURSOR FOR TRANSMISSIO
6497    17A9    2A   C0   FF         LHLD CURROW    ;GET NEW ROW AND COLUMN
6498    17AC    C1   .    .          POP  B         ;RECALL OLD ROW AND COLUMN
6499    17AD    7C   .    .          MOV  A,H       ;COMPARE TO HOME ROW AND
6500    17AE    90   .    .          SUB  B           ;COLUMN
6501    17AF    47   .    .          MOV  B,A
6502    17B0    7D   .    .          MOV  A,L
6503    17B1    91   .    .          SUB  C
6504    17B2    B0   .    .          ORA  B         ;DID CURSOR MOVE?
6505    17B3    C2   99   17         JNZ  STB060    ;YES - SET FOR DATA PRESENT
6506    17B6    3E   04   .          MVI  A,NOSEND  ;NO - SET FOR NO DATA
6507    17B8    C3   11   17         JMP  SETDFL    ;RETURN
```

```
=======================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=======================================================================
 6509   17BB   .   .   .   ;*****************************
 6510   17BB   .   .   .   ; XPUTDC - TRANSMIT CHARACTER *
 6511   17BB   .   .   .   ;*****************************
 6512   17BB   .   .   .   ;
 6513   17BB   .   .   .   ;  ENTRY:  A = CHARACTER TO BE TRANSMITTED
 6514   17BB   .   .   .   ;
 6515   17BB   .   .   .   ;  EXIT :  NC - TRANSMIT SUCCESSFUL
 6516   17BB   .   .   .   ;          C - TRANSMIT FAILED
 6517   17BB   .   .   .   ;          A DESTROYED
 6518   17BB   .   .   .   ;
 6519   17BB   .   .   .   ESCOUT EQU  $          ;OUTPUT AN ESCAPE CODE
 6520   17BB   3E  1B  .          MVI  A,ESC
 6521   17BD   CD  C1  17         CALL XPUTDC
 6522   17C0   78  .   .          MOV  A,B        ;FOLLOWED BY CHAR IN B-REG
 6523   17C1   .   .   .   ;
 6524   17C1   .   .   .   XPUTDC EQU  $
 6525   17C1   B7  .   .          ORA  A          ;SET C-FLAG FALSE
 6526   17C2   F5  .   .          PUSH PSW        ;SAVE THE FLAGS AND A-REG
 6527   17C3   CD  6A  10         CALL CKRMIE     ;REMOTE MODE ENABLED?
 6528   17C6   CA  D4  17         JZ   XPD005     ;NO - EXIT
 6529   17C9   .   .   .   XPD001 EQU  $
 6530   17C9   F1  .   .          POP  PSW        ;YES - RECALL THE CHARACTER
 6531   17CA   F5  .   .          PUSH PSW        ;SAVE CONTENTS OF A AND FLAG
 6532   17CB   CD  1A  50         CALL ZPUTDC     ;TRANSMIT THE CHAR IN A-REG
 6533   17CE   DA  EA  17         JC   XPD050     ;ERROR - REPORT IT
 6534   17D1   C2  D6  17         JNZ  XPD010     ;WAIT - TRY AGAIN
 6535   17D4   .   .   .   XPD005 EQU  $
 6536   17D4   F1  .   .          POP  PSW        ;DONE - RECALL FLAGS AND CHA
 6537   17D5   C9  .   .          RET             ;RETURN
 6538   17D6   .   .   .   ;                        TRANSFER TRIGGER (SETS
 6539   17D6   .   .   .   ;                        FLAG TRUE)
 6540   17D6   .   .   .   ;************************************
 6541   17D6   .   .   .   ; WAIT FOR DATACOM - RETRY OPERATION *
 6542   17D6   .   .   .   ;************************************
 6543   17D6   .   .   .   XPD010 EQU  $
 6544   17D6   E5  .   .          PUSH H          ;SAVE THE REGISTERS
 6545   17D7   D5  .   .          PUSH D
 6546   17D8   C5  .   .          PUSH B
 6547   17D9   CD  86  15         CALL IOCTMN     ;MONITOR CARTRIDGE TAPES
 6548   17DC   3E  0A  .          MVI  A,CKBRKY   ;LOOK FOR A BREAK KEY HIT
 6549   17DE   CD  08  48         CALL ZKBCTL     ;BREAK.KEY HIT?
 6550   17E1   C1  .   .          POP  B            ;(RESTORE REGISTERS)
 6551   17E2   D1  .   .          POP  D
 6552   17E3   E1  .   .          POP  H
 6553   17E4   CA  C9  17         JZ   XPD001     ;NO - TRY TO OUTPUT AGAIN
 6554   17E7   CD  3B  12         CALL BRKDC      ;YES - BREAK DATA COMM
 6555   17EA   .   .   .   ;                      FALL INTO ERROR EXIT ROUTINE
```

```
=============================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=============================================================================
6557   17EA   .   .   .   ;*****************************************
6558   17EA   .   .   .   ; DATA COMM ERROR DETECTED - REPORT ERROR *
6559   17EA   .   .   .   ;*****************************************
6560   17EA   .   .   .   XPD050 EQU  $
6561   17EA   33  .   .          INX  SP        ;RESTORE STACK LEVEL WITHOUT
6562   17EB   33  .   .          INX  SP          ;AFFECTING THE FLAGS
6563   17EC   C2  54  12         JNZ  HANGU0    ;FATAL - HANG THE TERMINAL
6564   17EF   CD  14  48         CALL ZBELL     ;NON-FATAL - SOUND BELL
6565   17F2   37  .   .          STC            ;RETURN FAIL (C-FLAG = TRUE)
6566   17F3   C9  .   .          RET
```

```
================================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                         PAGE 194
================================================================================
 6568   17F4    .    .    .   ;******************************************
 6569   17F4    .    .    .   ; XMOHME - HOME CURSOR INCLUDING TRANSMIT *
 6570   17F4    .    .    .   ;   ONLY FIELDS                           *
 6571   17F4    .    .    .   ;******************************************
 6572   17F4    .    .    .   XMOHME EQU  $
 6573   17F4    CD   OF   17         CALL SETDFO    ;SET DATA COMM INPUT FLAG
 6574   17F7    .    .    .   ;                       TO ENABLE TRANSMIT ONLY
 6575   17F7    .    .    .   ;                       FIELDS
 6576   17F7    C3   2C   1D         JMP  CURPH1    ;HOME THE CURSOR
```

```
6578   17FA    .   .   .   ;*************************
6579   17FA    .   .   .   ;  R O M   B R E A K   3  *
6580   17FA    .   .   .   ;*************************
6581   17FA    .   .   .           ORG   ZBRK2+4000Q
6582   1800    .   .   .   ZBRK3   EQU   $
6583   1800   50   .   .           DB    VERSN      ;ROM PRESENT FLAGS
6584   1801   18   .   .           DB    ZBRK3/256
```

```
 6586    1802   .    .    .    ;*************************************
 6587    1802   .    .    .    ; BACKT1 - LOCATE PREVIOUS CHARACTER *
 6588    1802   .    .    .    ;*************************************
 6589    1802   .    .    .    ;
 6590    1802   .    .    .    ;  ENTRY:  IODATA = CHARS TO FIND (2 BYTES)
 6591    1802   .    .    .    ;          CURCOL,CURROW = CURRENT CURSOR POSITION
 6592    1802   .    .    .    ;
 6593    1802   .    .    .    ;  EXIT :  Z - CHARACTER FOUND
 6594    1802   .    .    .    ;            DISPLAY AND CURSOR SET TO CHARACTER
 6595    1802   .    .    .    ;              POSITION IN DISPLAY MEMORY - ALL
 6596    1802   .    .    .    ;              DISPLAY PARAMETERS UPDATED
 6597    1802   .    .    .    ;          NZ - CHARACTER NOT FOUND
 6598    1802   .    .    .    ;            DISPLAY UNCHANGED
 6599    1802   .    .    .    ;          ALL REGISTERS DESTROYED
 6600    1802   .    .    .    ;
 6601    1802   .    .    .    BACKT0 EQU  $           ;LOOK FOR PREVIOUS FIELD
 6602    1802   21   C1   C1          LXI  H,ENDPR*256+ENDPR
 6603    1805   .    .    .    BACKT1 EQU  $
 6604    1805   22   D7   FF          SHLD LCHKSM      ;SAVE CHARACTERS TO BE FOUND
 6605    1808   AF   .    .           XRA  A           ;CLEAR ROLL COUNT
 6606    1809   32   82   FF          STA  ROLLCT
 6607    180C   2A   C0   FF          LHLD CURROW    ;SAVE THE CURRENT STATE OF
 6608    180F   22   DE   FF          SHLD LDATA       ;THE DISPLAY
 6609    1812   2A   C9   FF          LHLD LSTLIN
 6610    1815   22   D5   FF          SHLD LADDR
 6611    1818   3E   01   .           MVI  A,IGNTRM    ;SET TO IGNORE NON-DISPLAYIN
 6612    181A   32   6D   FF          STA  TRMFCT        ;TERMINATOR
 6613    181D   CD   B4   06          CALL RCADR1      ;DOES THE CURRENT LINE EXITS
 6614    1820   3A   DF   FF          LDA  LDATA+1       ;(RECALL CURRENT COLUMN)
 6615    1823   F2   3B   18          JP   BKT230      ;YES - SEARCH FOR PREV FIELD
 6616    1826   3A   C0   FF          LDA  CURROW      ;NO - LOCATE LAST LINE
 6617    1829   21   6B   FF          LXI  H,MLKROW    ;CURRENT ROW LESS THAN
 6618    182C   BE   .    .           CMP  M             ;MEMORY LOCK ROW?
 6619    182D   F2   7A   18          JP   BKT300      ;NO - LOOK FOR UNLOCKED LINE
 6620    1830   .    .    .    BKT210 EQU  $           ;YES - START FROM LAST LINE
 6621    1830   CD   07   11          CALL CURPHD
 6622    1833   3A   C7   FF          LDA  LSTROW      ;FORCE TO LAST ALLOCATED
 6623    1836   32   C0   FF          STA  CURROW        ;ROW
```

```
=====================================================================
ITEM   LOC   OBJECT CODE  SOURCE STATEMENTS                  PAGE 197
=====================================================================
6625   1839   .   .   .   ;************************************
6626   1839   .   .   .   ; LOCATE THE LAST FIELD IN THE LINE *
6627   1839   .   .   .   ;************************************
6628   1839   .   .   .   BKT220 EQU  $
6629   1839   3E  4F  .          MVI  A,MAXCOL  ;SET SEARCH LIMIT
6630   183B   .   .   .   BKT230 EQU  $
6631   183B   32  85  FF         STA  TMPCOL  ;SAVE THE SEARCH LIMIT
6632   183E   2A  C9  FF         LHLD LSTLIN  ;GET SEARCH START ADDRESS
6633   1841   EB  .   .          XCHG           ;PUT ADDRESS IN D,E
6634   1842   2A  D7  FF         LHLD LCHKSM    ;RECALL CHARS TO BE FOUND
6635   1845   CD  5C  1F         CALL FNDLST    ;ANY FIELDS IN LINE?
6636   1848   F2  90  18         JP   BKT400    ;YES - SET DISPLAY TO FIELD
6637   184B   3A  6B  FF         LDA  MLKROW    ;NO - SEE IF TOP UNLOCKED
6638   184E   21  C0  FF         LXI  H,CURROW    ;LINE HAS BEEN REACHED
6639   1851   BE  .   .          CMP  M         ;REACHED MEMORY LOCK ROW?
6640   1852   CA  86  18         JZ   BKT310    ;YES - CONTINUE ABOVE DISPLA
6641   1855   3A  82  FF         LDA  ROLLCT
6642   1858   B6  .   .          ORA  M         ;ROLL COUNT AND ROW = ZERO?
6643   1859   CA  F0  18         JZ   BKT500    ;YES - NO PREVIOUS FIELD IN
6644   185C   .   .   .   ;                       LOCKED AREA, RESTORE DISPL
6645   185C   35  .   .          DCR  M         ;NO - MOVE TO PREVIOUS ROW
6646   185D   2E  82  .          MVI  L,ROLLCT
6647   185F   7E  .   .          MOV  A,M       ;GET ROLL COUNT
6648   1860   B7  .   .          ORA  A         ;SEARCHING ABOVE DISPLAY?
6649   1861   CA  68  18         JZ   BKT240    ;NO - DON'T INCREMENT COUNT
6650   1864   34  .   .          INR  M         ;ROLL OVERFLOW?
6651   1865   CA  F0  18         JZ   BKT500    ;YES - RESTORE DISPLAY
6652   1868   .   .   .   BKT240 EQU  $         ;NO - LOOK TO PREVIOUS LINE
6653   1868   2A  C9  FF         LHLD LSTLIN  ;RECALL CURRENT LINE ADDR
6654   186B   .   .   .   BKT250 EQU  $
6655   186B   23  .   .          INX  H         ;GET ADDRESS OF PREVIOUS
6656   186C   23  .   .          INX  H           ;LINE
6657   186D   CD  6D  19         CALL CHAIN     ;GET PREVIOUS LINE ADDRESS
6658   1870   B7  .   .          ORA  A         ;DOES PREVIOUS LINE EXIST?
6659   1871   CA  F0  18         JZ   BKT500    ;NO - RESTORE DISPLAY
6660   1874   22  C9  FF         SHLD LSTLIN    ;YES - SAVE ADDRESS OF LINE
6661   1877   C3  39  18         JMP  BKT220    ;LOCATE LAST FIELD IN LINE
```

```
 6663   187A   .    .    .   ;
 6664   187A   .    .    .   ;  ROW NOT FOUND AND CURSOR BELOW MEMORY LOCK
 6665   187A   .    .    .   ;    LINE - LOCATE LAST LINE TO START
 6666   187A   .    .    .   ;
 6667   187A   .    .    .   BKT300 EQU  $
 6668   187A   93   .    .          SUB  E              ;(E = # OF ROWS TO LAST LN
 6669   187B   BE   .    .          CMP  M            ;LAST ROW BELOW LOCKED AREA?
 6670   187C   F2   30   18         JP   BKT210       ;YES - START AT LAST LINE
 6671   187F   7E   .    .          MOV  A,M          ;NO - SEARCH ABOVE DISPLAY
 6672   1880   32   C0   FF         STA  CURROW       ;SET "CURROW" TO MEM LOCK RO
 6673   1883   21   C0   FF         LXI  H,CURROW     ;SET H,L -> "CURROW"
 6674   1886   .    .    .   ;
 6675   1886   .    .    .   ;  NO PREVIOUS FIELDS ON DISPLAY - LOOK ABOVE DISP
 6676   1886   .    .    .   ;
 6677   1886   .    .    .   BKT310 EQU  $
 6678   1886   35   .    .          DCR  M            ;DECREMENT ROW NUMBER
 6679   1887   2E   82   .          MVI  L,ROLLCT-BASE
 6680   1889   34   .    .          INR  M            ;INCREMENT ROLL COUNT
 6681   188A   2A   CB   FF         LHLD TOPLIN     ;GET TOP DISPLAY LINE ADDR
 6682   188D   C3   6B   18         JMP  BKT250       ;LOOK FOR PREVIOUS ROW
```

```
6684   1890   .   .   .   ;
6685   1890   .   .   .   ;  FIELD FOUND - SET DISPLAY
6686   1890   .   .   .   ;
6687   1890   .   .   .   BKT400 EQU  $
6688   1890   2A  D5  FF         LHLD LADDR      ;RESTORE VALUE OF LSTLIN
6689   1893   EB  .   .          XCHG              ;AND SAVE ADDRESS OF
6690   1894   2A  C9  FF         LHLD LSTLIN       ;LINE WHERE FIELD IS
6691   1897   22  D5  FF         SHLD LADDR
6692   189A   EB  .   .          XCHG
6693   189B   22  C9  FF         SHLD LSTLIN
6694   189E   3A  85  FF         LDA  TMPCOL  ;COMPUTE COLUMN LOCATION
6695   18A1   90  .   .          SUB  B
6696   18A2   32  C1  FF         STA  CURCOL
6697   18A5   3A  82  FF         LDA  ROLLCT
6698   18A8   B7  .   .          ORA  A          ;ROW ABOVE DISPLAY?
6699   18A9   CA  DB  18         JZ   BKT450     ;NO - EXIT
6700   18AC   .   .   .   ;
6701   18AC   .   .   .   ;  ROW ABOVE DISPLAY ROLL IT DOWN
6702   18AC   .   .   .   ;
6703   18AC   .   .   .   BKT410 EQU  $
6704   18AC   3E  E8  .          MVI  A,-MAXROW-1  ;COMPUTE NUMBER OF LINES
6705   18AE   21  6B  FF         LXI  H,MLKROW  ;TO ROLL FOR ONE PAGE
6706   18B1   86  .   .          ADD  M
6707   18B2   32  82  FF         STA  ROLLCT  ;SAVE ROLL COUNT
6708   18B5   .   .   .   BKT420 EQU  $
6709   18B5   CD  C5  0B         CALL ROLLDN     ;ROLL DOWN ONE LINE
6710   18B8   CA  D1  18         JZ   BKT430     ;ROLL FAIL - CHECK FOR FIELD
6711   18BB   21  C0  FF         LXI  H,CURROW
6712   18BE   34  .   .          INR  M          ;INCREMENT ROW NUMBER
6713   18BF   2E  82  .          MVI  L,ROLLCT
6714   18C1   34  .   .          INR  M          ;PAGE COMPLETED?
6715   18C2   C2  B5  18         JNZ  BKT420     ;NO - CONTINUE ROLLING
6716   18C5   3A  C0  FF         LDA  CURROW
6717   18C8   2E  6B  .          MVI  L,MLKROW
6718   18CA   96  .   .          SUB  M          ;IS DESIRED ROW ON SCREEN?
6719   18CB   FA  AC  18         JM   BKT410     ;NO - ROLL DOWN ANOTHER PAGE
6720   18CE   C3  DB  18         JMP  BKT450     ;YES - EXIT
6721   18D1   .   .   .   ;
6722   18D1   .   .   .   ;  ROLL FAILED - CHECK FOR FIELD ON SCREEN
6723   18D1   .   .   .   ;
6724   18D1   .   .   .   BKT430 EQU  $
6725   18D1   3A  C0  FF         LDA  CURROW  ;GET CURRENT ROW NUMBER
6726   18D4   21  6B  FF         LXI  H,MLKROW  ;SUBTRACT MEMRY LOCK RWS
6727   18D7   96  .   .          SUB  M          ;IS FIELD ON SCREEN?
6728   18D8   FA  F6  18         JM   BKT510     ;NO - RESTORE DISPLAY, ROLL
6729   18DB   .   .   .   ;                          DOWN FAILED BECAUSE OF NO
6730   18DB   .   .   .   ;                          MEMORY TO FILL TO MEMORY
6731   18DB   .   .   .   ;                          LOCK LINE
```

```
 6733    18DB    .    .    .   ;
 6734    18DB    .    .    .   ;   FIELD ON SCREEN - SET SCREEN VALUES
 6735    18DB    .    .    .   ;
 6736    18DB    .    .    .   BKT450 EQU   $
 6737    18DB    3A   C0   FF         LDA   CURROW  ;SET LAST ROW VALUE TO
 6738    18DE    32   C7   FF         STA   LSTROW    ;ROW FOUND
 6739    18E1    AF   .    .          XRA   A         ;SET LAST COL DONE TO ZERO
 6740    18E2    32   C8   FF         STA   LSTCOL
 6741    18E5    2A   D5   FF         LHLD  LADDR     ;SET ADDRESSES TO LOCATION
 6742    18E8    .    .    .   BACKI5 EQU   $
 6743    18E8    22   C9   FF         SHLD  LSTLIN      ;OF FIELD
 6744    18EB    2B   .    .          DCX   H         ;SET CURADR TO CORRESPOND
 6745    18EC    22   C3   FF         SHLD  CURADR
 6746    18EF    C9   .    .          RET             ;RETURN
 6747    18F0    .    .    .   ;
 6748    18F0    .    .    .   ;   FIELD NOT FOUND - RESTORE DISPLAY
 6749    18F0    .    .    .   ;
 6750    18F0    .    .    .   BKT500 EQU   $
 6751    18F0    2A   D5   FF         LHLD  LADDR   ;RESTORE LAST LINE ADDRESS
 6752    18F3    22   C9   FF         SHLD  LSTLIN
 6753    18F6    .    .    .   BKT510 EQU   $
 6754    18F6    2A   DE   FF         LHLD  LDATA   ;RESTORE CURRENT ROW AND
 6755    18F9    22   C0   FF         SHLD  CURROW    ;COLUMN
 6756    18FC    F6   FF   .          ORI   377Q      ;SET Z FALSE
 6757    18FE    C9   .    .          RET             ;RETURN NOT FOUND (NZ)
```

```
==========================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                     PAGE 201
==========================================================================
6759    18FF   .    .    .   ;*********************
6760    18FF   .    .    .   ;  BKTAB - BACK TAB  *
6761    18FF   .    .    .   ;*********************
6762    18FF   .    .    .   BKTAB   EQU  $
6763    18FF   CD   72   19          CALL CHKFMO     ;FORMAT/SOFT KEY DEFINE MODE
6764    1902   C2   02   18          JNZ  BACKTO     ;YES - LOCATE PREVIOUS FIELD
6765    1905   3A   C1   FF          LDA  CURCOL     ;NO - FIND PREVIOUS SET TAB
6766    1908   3D   .    .           DCR  A          ;START AT PREVIOUS COLUMN
6767    1909   2E   BF   .           MVI  L,LFTMGN-BASE
6768    190B   BE   .    .           CMP  M          ;WHERE IS CURSOR?
6769    190C   CA   98   11          JZ   CURP04     ;AT MARGIN - SET DISPLAY
6770    190F   F2   39   19          JP   BKT100     ;AFTER MARGIN - FIND PREV TA
6771    1912   .    .    .   ;
6772    1912   .    .    .   ;  CURSOR AT BEGINNING OF LINE - LOCATE TAB IN
6773    1912   .    .    .   ;    PREVIOUS LINE
6774    1912   .    .    .   ;
6775    1912   3A   6B   FF          LDA  MLKROW  ;GET MEMORY LOCK ROW
6776    1915   2E   C0   .           MVI  L,CURROW
6777    1917   BE   .    .           CMP  M          ;CURRENT ROW = LOCK ROW?
6778    1918   C2   22   19          JNZ  BKT010     ;NO - MOVE CURSOR UP ONE ROW
6779    191B   CD   C5   0B          CALL ROLLDN     ;YES - ROLL DOWN ONE LINE
6780    191E   C8   .    .           RZ              ;CAN'T ROLL DOWN - EXIT
6781    191F   C3   26   19          JMP  BKT050     ;GO LOCATE LAST TAB SET
6782    1922   .    .    .   ;
6783    1922   .    .    .   ;  CURSOR NOT AT TOP OF FREE AREA - MOVE UP 1 LINE
6784    1922   .    .    .   ;
6785    1922   .    .    .   BKT010  EQU  $
6786    1922   7E   .    .           MOV  A,M        ;GET CURRENT ROW NUMBER
6787    1923   B7   .    .           ORA  A          ;ROW = 0
6788    1924   C8   .    .           RZ              ;YES - DON'T BACK TAB WHEN
6789    1925   .    .    .   ;                         CURSOR IS LOCATED IN ROW
6790    1925   .    .    .   ;                         ZERO AND DISPLAY LOCK ON
6791    1925   35   .    .           DCR  M          ;NO - DECREMENT ROW NUMBER
```

```
 6793   1926   .    .    .   ;
 6794   1926   .    .    .   ;  PREVIOUS ROW LOCATED - LOCATE LAST TAB SET
 6795   1926   .    .    .   ;
 6796   1926   .    .    .   BKT050 EQU   $
 6797   1926   3A   81   FF         LDA   HTBTBL+9   ;GET LAST TAB ENTRY
 6798   1929   E6   80   .          ANI   200Q       ;LAST TAB SET?
 6799   192B   3E   4F   .          MVI   A,MAXCOL     ;(SET FOR LAST COLUMN-1)
 6800   192D   CA   38   19         JZ    BKT060     ;NO - LOCATE LAST TAB
 6801   1930   3C   .    .          INR   A          ;YES - SET FOR LAST COLUMN #
 6802   1931   4F   .    .          MOV   C,A
 6803   1932   2E   BE   .          MVI   L,RHTMGN-BASE
 6804   1934   BE   .    .          CMP   M          ;RIGHT MARGIN = LAST COLUMN?
 6805   1935   CA   98   11         JZ    CURP04     ;YES - SET CURSOR TO LAST CO
 6806   1938   .    .    .   BKT060 EQU   $          ;NO - SET TO MAXCOL-1 AND
 6807   1938   3D   .    .          DCR   A            ;LOCATE PREVIOUS TAB
 6808   1939   .    .    .   ;
 6809   1939   .    .    .   ;  LOCATE PREVIOUS TAB (A = CURRENT COLUMN - 1)
 6810   1939   .    .    .   ;
 6811   1939   .    .    .   BKT100 EQU   $
 6812   1939   3C   .    .          INR   A          ;RESTORE CURRENT COLUMN
 6813   193A   47   .    .          MOV   B,A        ;SAVE IT
 6814   193B   F6   07   .          ORI   7Q         ;SET TO COLUMN CORRESPOINDIN
 6815   193D   .    .    .   ;                          TO LAST BIT OF TAB BYTE
 6816   193D   2E   BF   .          MVI   L,LFTMGN-BASE
 6817   193F   96   .    .          SUB   M          ;COMPUTE NUMBER OF CHARS
 6818   1940   4F   .    .          MOV   C,A          ;TO SEARCH
 6819   1941   78   .    .          MOV   A,B        ;RECALL CURRENT COLUMN
 6820   1942   CD   ED   14         CALL  FNDTB1     ;GET BYTE MASK AND
 6821   1945   .    .    .   ;                          CORRESPONDING TABLE BYTE
 6822   1945   3D   .    .          DCR   A          ;SET FOR MASK TO MASK OFF
 6823   1946   A6   .    .          ANA   M            ;SUCCEEDING TABS
 6824   1947   .    .    .   ;
 6825   1947   .    .    .   ;  LOCATE PREVIOUS TAB SETTING
 6826   1947   .    .    .   ;
 6827   1947   .    .    .   BKT110 EQU   $
 6828   1947   06   08   .          MVI   B,8        ;INITIALIZE BIT COUNT
 6829   1949   .    .    .   BKT120 EQU   $
 6830   1949   07   .    .          RLC              ;TAB SET?
 6831   194A   D2   5B   19         JNC   BKT150     ;NO - BACK UP ANOTHER COLUMN
 6832   194D   .    .    .   ;
 6833   194D   .    .    .   ;  TAB LOCATED - SET CURSOR (C = TAB COLUMN)
 6834   194D   .    .    .   ;
 6835   194D   .    .    .   BKT130 EQU   $
 6836   194D   5F   .    .          MOV   E,A        ;SAVE A-REGISTER
 6837   194E   79   .    .          MOV   A,C        ;PUT COLUMN NUMBER IN A-REG
 6838   194F   E5   .    .          PUSH  H          ;SAVE H AND L
 6839   1950   2A   BE   FF         LHLD  RHTMGN     ;GET MARGIN SETTINGS
 6840   1953   84   .    .          ADD   H          ;COMPUTE TAB COLUMN LOCATION
 6841   1954   2C   .    .          INR   L          ;IS TAB LOCATION BEYOND LEFT
 6842   1955   BD   .    .          CMP   L            ;MARGIN?
 6843   1956   E1   .    .          POP   H            ;(RESTORE H AND L)
 6844   1957   FA   98   11         JM    CURP04     ;NO - LOCATE TAB AND RETURN
```

```
=============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                     PAGE 203
=============================================================================
 6845    195A   7B  .   .          MOV  A,E       ;YES - RECALL A-REGISTER
```

```
==============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                        PAGE 204
==============================================================================
 6847   195B    .   .   .   ;******************************
 6848   195B    .   .   .   ; CONTINUE SCANNING BACKWARDS *
 6849   195B    .   .   .   ;******************************
 6850   195B    .   .   .   BKT150 EQU  $
 6851   195B   0D   .   .          DCR  C          ;COLUMN ZERO REACHED?
 6852   195C   CA  4D  19          JZ   BKT130     ;YES - SET CURSOR COLUMN
 6853   195F   05   .   .          DCR  B          ;BYTE DONE?
 6854   1960   C2  49  19          JNZ  BKT120     ;NO - CONTINUE TO NEXT COLUM
 6855   1963   2B   .   .          DCX  H          ;YES - GET NEXT BYTE
 6856   1964   7E   .   .          MOV  A,M
 6857   1965   C3  47  19          JMP  BKT110     ;CHECK BYTE FOR TAB SET
```

```
=====================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                               PAGE 205
=====================================================================================
 6859   1968    .    .    .  ;****************************************
 6860   1968    .    .    .  ; CHAIN - SET H,L TO POINTER FROM MEMORY *
 6861   1968    .    .    .  ;****************************************
 6862   1968    .    .    .  ;
 6863   1968    .    .    .  ;  ENTRY:  H,L = ADDRESS OF POINTER
 6864   1968    .    .    .  ;
 6865   1968    .    .    .  ;  EXIT :  A = LSB OF POINTER
 6866   1968    .    .    .  ;          H,L = POINTER VALUE
 6867   1968    .    .    .  ;
 6868   1968    .    .    .  CHAIN0 EQU  $
 6869   1968   EB    .    .         XCHG               ;PUT ADDRESS INTO H,L
 6870   1969    .    .    .  CHAIN1 EQU  $
 6871   1969   7D    .    .         MOV  A,L           ;COMPUTE LOCATION OF NEXT
 6872   196A   E6   F0    .         ANI  377Q-BLKSM  ;BLOCK POINTEP IN BLOCK
 6873   196C   6F    .    .         MOV  L,A           ;GET THE NEXT BLOCK ADDRESS
 6874   196D    .    .    .  CHAIN  EQU  $
 6875   196D   7E    .    .         MOV  A,M           ;GET LSB OF POINTER
 6876   196E   23    .    .         INX  H
 6877   196F   66    .    .         MOV  H,M           ;PUT MSB INTO H-REGISTEP
 6878   1970   6F    .    .         MOV  L,A           ;PUT LSB INTO L-REGISTER
 6879   1971    .    .    .  NOFNCT EQU  $               ;(NON-FUNCTION FOP ESC SEQ
 6880   1971   C9    .    .         RET                ;RETURN
 6882   1972    .   .   .   ;************************************************
 6883   1972    .   .   .   ; CHKFMS - CHECK FORMAT AND SOFT KEY DEFINE MODE *
 6884   1972    .   .   .   ;************************************************
 6885   1972    .   .   .   ;
 6886   1972    .   .   .   ;  ENTRY:  DON'T CARE
 6887   1972    .   .   .   ;
 6888   1972    .   .   .   ;  EXIT :  Z - NEITHER MODE ENABLED
 6889   1972    .   .   .   ;            A = 0
 6890   1972    .   .   .   ;          NZ - MODE ENABLED
 6891   1972    .   .   .   ;            A = -1, SOFT KEY MODE ENABLED
 6892   1972    .   .   .   ;            A >  0, FORMAT MODE ONLY ENABLED
 6893   1972    .   .   .   ;
 6894   1972    .   .   .   CHKFMO EQU   $
 6895   1972    2E  6C  .          MVI   L,SPOWL     ;TURN OF SPOW LATCH FIRST
 6896   1974    36  FF  .          MVI   M,SPOWOF
 6897   1976    .   .   .   CHKFMS EQU  $
 6898   1976    3A  AE  FF         LDA   DSPTYP      ;GET DISPLAY TYPE FLAG
 6899   1979    B7  .   .          ORA   A           ;SOFT KEY DISPLAY ON?
 6900   197A    C0  .   .          RNZ               ;YES - RETURN
 6901   197B    .   .   .   CHKFMT EQU   $
 6902   197B    3A  F4  FF         LDA   MDFLG1      ;NO - GET MODE FLAGS
 6903   197E    E6  08  .          ANI   FORMAT      ;MASK FOR FORMAT FLAG AND
 6904   1980    C9  .   .          RET                  ;RETURN
```

```
6906   1981   .   .   .   ;****************************************
6907   1981   .   .   .   ; CHKMLK - CHECK FOR MEMORY LOCK ENABLED *
6908   1981   .   .   .   ;****************************************
6909   1981   .   .   .   ;
6910   1981   .   .   .   ;  ENTRY:  DON'T CARE
6911   1981   .   .   .   ;
6912   1981   .   .   .   ;  EXIT :  Z => MEMORY LOCK ENABLED
6913   1981   .   .   .   ;          NZ => MEMORY LOCK NOT ENABLED
6914   1981   .   .   .   ;          A,H,L DESTROYED
6915   1981   .   .   .   ;
6916   1981   .   .   .   CHKMLK EQU  $
6917   1981   3A  F4  FF         LDA  MDFLG1     ;GET SOFT MODE FLAGS
6918   1984   2F  .   .          CMA             ;MEMORY LOCK ENABLED FOR FUL
6919   1985   E6  04  .          ANI  MEMLOK       ;LOCKOUT IF MEMORY LOCK SE
6920   1987   21  6B  FF         LXI  H,MLKROW     ;AND MEMORY LOCK ROW = 0
6921   198A   B6  .   .          ORA  M
6922   198B   C9  .   .          RET             ;RETURN
6923   198C   .   .   .   ;*********************************
6924   198C   .   .   .   ; CHKSFK - CHECK FOR SOFT KEY MODE *
6925   198C   .   .   .   ;*********************************
6926   198C   .   .   .   ;
6927   198C   .   .   .   ;  EXIT :  Z - NORMAL MODE
6928   198C   .   .   .   ;            A = 0
6929   198C   .   .   .   ;          NZ - SOFT KEY DEFINE MODE
6930   198C   .   .   .   ;            A DESTROYED
6931   198C   .   .   .   ;
6932   198C   .   .   .   CHKSFK EQU  $
6933   198C   3A  AE  FF         LDA  DSPTYP     ;GET DISPLAY TYPE FLAG
6934   198F   B7  .   .          ORA  A          ;SET Z FALSE IF SOFT KEY
6935   1990   C9  .   .          RET               ;ON AND RETURN
```

```
 6937    1991    .    .    .   ;*************************
 6938    1991    .    .    .   ; CD - CHARACTER DELETE *
 6939    1991    .    .    .   ;*************************
 6940    1991    .    .    .   DELWRP EQU  $            ;DELETE WITH WRAP AROUND
 6941    1991    CD   76   19         CALL CHKFMS       ;FORMAT/SOFT KEY DEFINE MODE
 6942    1994    3E   20   .          MVI  A,WRPDEL       ;(PUT WRAP FLAG IN A-REG)
 6943    1996    CC   39   17         CZ   SETMF2       ;NO - SET WRAP AROUND FLAG
 6944    1999    .    .    .   ;
 6945    1999    .    .    .   CHRDEL EQU  $
 6946    1999    CD   8C   19         CALL CHKSFK       ;SOFT KEY DEFINE MODE?
 6947    199C    CA   A4   19         JZ   CHD010       ;NO - DO DELETE
 6948    199F    3A   C0   FF         LDA  CURROW       ;YES - GET CURSOR ROW
 6949    19A2    0F   .    .          RRC               ;IN DATA LINE?
 6950    19A3    D0   .    .          RNC               ;NO - RETURN
 6951    19A4    .    .    .   CHD010 EQU  $            ;YES - DO DELETE
 6952    19A4    AF   .    .          XRA  A            ;ZERO SAVE AREA
 6953    19A5    32   98   FF         STA  CHSAV
 6954    19A8    CD   0A   1A         CALL CHD000       ;DELETE A CHARACTER
 6955    19AB    3A   98   FF         LDA  CHSAV        ;RECALL THE DELETED CHARACTE
 6956    19AE    B7   .    .          ORA  A            ;WAS IT A DISPLAY CONTROL?
 6957    19AF    FA   99   19         JM   CHRDEL       ;YES - CONTINUE DELETING
 6958    19B2    .    .    .   ;*******************************************
 6959    19B2    .    .    .   ; ADJUST FOR CHARACTERS BEYOND RIGHT MARGIN *
 6960    19B2    .    .    .   ;*******************************************
 6961    19B2    21   C1   FF         LXI  H,CURCOL
 6962    19B5    3A   BE   FF         LDA  RHTMGN
 6963    19B8    BE   .    .          CMP  M            ;CURSOR BEYOND RIGHT MARGIN?
 6964    19B9    D8   .    .          RC                ;YES - DON'T CHECK WRAP
 6965    19BA    46   .    .          MOV  B,M          ;NO - SAVE CURRENT COLUMN
 6966    19BB    77   .    .          MOV  M,A          ;SET COLUMN TO RIGHT MARGIN
 6967    19BC    57   .    .          MOV  D,A          ;SAVE RIGHT MARGIN VALUE
 6968    19BD    2E   F4   .          MVI  L,MDFLG1-BASE
 6969    19BF    4E   .    .          MOV  C,M          ;SAVE SOFT MODE FLAGS STATE
 6970    19C0    C5   .    .          PUSH B              ;AND CURRENT COLUMN
 6971    19C1    79   .    .          MOV  A,C          ;FORCE THE INSERT CHARACTER
 6972    19C2    E6   FD   .          ANI  377Q-INSCHR  ;MODE OFF
 6973    19C4    77   .    .          MOV  M,A
```

```
==========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                      PAGE 209
==========================================================================
 6975   19C5    .    .    .   ;*******************************************
 6976   19C5    .    .    .   ; DELETE PERFORMED - CHECK FOR WRAP AROUND *
 6977   19C5    .    .    .   ;*******************************************
 6978   19C5   21   6F   FF          LXI  H,MFLGS2  ;GET TERMINAL MODE FLAGS
 6979   19C8   7E    .    .          MOV  A,M       ;MASK OUT DELETE WRAP FLAG
 6980   19C9   E6   DF    .          ANI  377Q-WRPDEL
 6981   19CB   BE    .    .          CMP  M         ;DELETE WRAP AROUND ENABLED?
 6982   19CC   CA   01   1A          JZ   CHD050    ;NO - EXIT
 6983   19CF   77    .    .          MOV  M,A       ;YES - UPDATE MODE FLAGS
 6984   19D0    .    .    .   ;********************************************
 6985   19D0    .    .    .   ; TRANSFER A CHRACTER UP FROM THE NEXT LINE *
 6986   19D0    .    .    .   ;********************************************
 6987   19D0    .    .    .   CHD020 EQU  $
 6988   19D0   3E   20    .          MVI  A,ABLNK   ;PRESET DELETED CHARACTER
 6989   19D2   32   98   FF          STA  CHSAV       ;TO A BLANK
 6990   19D5   21   C0   FF          LXI  H,CURROW  ;SET TO DELETE FIRST
 6991   19D8   34    .    .          INR  M           ;CHARACTER AT LEFT MARGIN
 6992   19D9   23    .    .          INX  H           ;FROM NEXT ROW
 6993   19DA   3A   BF   FF          LDA  LFTMGN
 6994   19DD   77    .    .          MOV  M,A
 6995   19DE   CD   CD   06          CALL RCADR4    ;CHARACTER EXIST?
 6996   19E1   CC   19   1A          CZ   CHRDL1    ;YES - DELETE IT
 6997   19E4   21   C0   FF          LXI  H,CURROW  ;RESTORE ROW NUMBER AND SET
 6998   19E7   35    .    .          DCR  M           ;COLUMN TO RIGHT MARGIN
 6999   19E8   23    .    .          INX  H
 7000   19E9   3A   BE   FF          LDA  RHTMGN
 7001   19EC   77    .    .          MOV  M,A
 7002   19ED   3A   98   FF          LDA  CHSAV     ;GET THE DELETED CHARACTER
 7003   19F0   FE   20    .          CPI  ABLNK     ;BLANK CHARACTER DELETED?
 7004   19F2   CA   01   1A          JZ   CHD050    ;YES - EXIT
 7005   19F5   06   00    .          MVI  B,0       ;NO - SET TO FORCE ENHANCE
 7006   19F7   CD   E2   21          CALL DISPC2    ;DISPLAY THE CHARACTER
 7007   19FA   3A   98   FF          LDA  CHSAV     ;RECALL THE DELETED CHARACTE
 7008   19FD   B7    .    .          ORA  A         ;WAS IT ASCII?
 7009   19FE   FA   D0   19          JM   CHD020    ;NO - TRANSFER ANOTHER BYTE
 7010   1A01    .    .    .   ;********************************************
 7011   1A01    .    .    .   ; EXIT - RESTORE CURSOR COLUMN AND "MDFLG1" *
 7012   1A01    .    .    .   ;********************************************
 7013   1A01    .    .    .   CHD050 EQU  $
 7014   1A01   C1    .    .          POP  B         ;RECALL ORIGINAL VALUES
 7015   1A02   21   C1   FF          LXI  H,CURCOL
 7016   1A05   70    .    .          MOV  M,B       ;RESTORE CURSOR COLUMN
 7017   1A06   2E   F4    .          MVI  L,MDFLG1-BASE
 7018   1A08   71    .    .          MOV  M,C       ;RESTORE "MDFLG1"
 7019   1A09   C9    .    .          RET            ;RETURN
```

```
==============================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                        PAGE 210
==============================================================================
 7021    1A0A    .    .    .   CHD000 EQU $
 7022    1A0A    CD   CD   06         CALL RCADR4    ;DOES CHARACTER EXIST?
 7023    1A0D    C0   .    .          RNZ            ;NO - RETURN
 7024    1A0E    CD   9A   1A         CALL CHD500    ;SKIP OVER SINGLE DISPLAY
 7025    1A11    .    .    .   ;                       ENHANCEMENT CODE
 7026    1A11    CD   7B   19         CALL CHKFMT    ;FORMAT MODE?
 7027    1A14    CA   19   1A         JZ   CHD100    ;NO - DELETE THE CHARACTER
 7028    1A17    04   .    .          INR  B         ;CURSOR IN PROTECTED FIELD?
 7029    1A18    C8   .    .          RZ             ;YES - RETURN
 7030    1A19    .    .    .   ;*********************************
 7031    1A19    .    .    .   ; CHRDL1 - DELETE ONE CHARACTER *
 7032    1A19    .    .    .   ;*********************************
 7033    1A19    .    .    .   ;
 7034    1A19    .    .    .   ;  ENTRY:  C = CHARACTER COLUMN POSITION
 7035    1A19    .    .    .   ;          D,E = ADDRESS OF CHAR TO BE DELETED
 7036    1A19    .    .    .   ;
 7037    1A19    .    .    .   ;  EXIT :  ALL REGISTERS DESTROYED
 7038    1A19    .    .    .   ;          CHSAV = CHARACTER DELETED (UNCHANGED
 7039    1A19    .    .    .   ;            IF A CHARACTER HAS NOT BEEN DELETED)
 7040    1A19    .    .    .   ;
 7041    1A19    .    .    .   CHRDL1 EQU  $
 7042    1A19    .    .    .   CHD100 EQU  $
 7043    1A19    1A   .    .          LDAX D         ;GET CHARACTER TO BE DELETED
 7044    1A1A    FE   CC   .          CPI EOL      ;IS IT EOL?
 7045    1A1C    C8   .    .          RZ           ;YES - RETURN
 7046    1A1D    32   98   FF         STA  CHSAV     ;SAVE THE DELETED CHARACTER
 7047    1A20    62   .    .          MOV  H,D       ;H,L = ADDR OF CHAR TO FILL
 7048    1A21    6B   .    .          MOV  L,E       ;D,E = ADDR OF CHAR TO MOVE
 7049    1A22    .    .    .   ;*********************************************
 7050    1A22    .    .    .   ; MOVE CHARACTERS DOWN TO PREVIOUS POSITION *
 7051    1A22    .    .    .   ;*********************************************
 7052    1A22    .    .    .   CHD110 EQU  $
 7053    1A22    CD   87   0B         CALL NXTCHR    ;GET THE NEXT CHARACTER
 7054    1A25    C2   74   1A         JNZ  CHD210    ;EOL LINK - TERMINATE DELETE
 7055    1A28    47   .    .          MOV  B,A       ;SAVE CHARACTER IN B-REGISTE
 7056    1A29    FE   C0   .          CPI  ENHLIM+1  ;ASCII OR ENHANCEMENT CODE?
 7057    1A2B    DA   49   1A         JC   CHD120    ;YES - SEE IF PAST MARGIN
 7058    1A2E    .    .    .   ;*********************************************
 7059    1A2E    .    .    .   ; FORMAT CONTROL CODE FOUND - CHECK FUNCTION *
 7060    1A2E    .    .    .   ;*********************************************
 7061    1A2E    FE   CC   .          CPI  EOL       ;END OF LINE?
 7062    1A30    CA   88   1A         JZ   CHD250    ;YES - TERMINATE DELETE
 7063    1A33    FE   C3   .          CPI  FILL      ;END OF LINE FILL?
 7064    1A35    CA   70   1A         JZ   CHD200    ;YES - TERMINATE DELETE
 7065    1A38    CD   91   1A         CALL CHD400    ;FORMAT MODE & DELETE ASCII?
 7066    1A3B    CA   49   1A         JZ   CHD120    ;NO - MOVE NEW CHARACTER
 7067    1A3E    78   .    .          MOV  A,B       ;YES - GET CHAR TO BE MOVED
 7068    1A3F    FE   C0   .          CPI  STPR      ;IS IT START PROTECT?
 7069    1A41    CA   9D   1C         JZ   CLER02    ;YES - CLEAR REST OF FIELD
 7070    1A44    .    .    .   ;                        AND TERMINATE DELETE
 7071    1A44    FE   C5   .          CPI  ALPHA     ;TYPE DEFINITION?
 7072    1A46    F2   22   1A         JP   CHD110    ;YES - SKIP OVER CHARACTER
 7074   1A49   .   .   .   ;***************************************
 7075   1A49   .   .   .   ; CHARACTER TO BE MOVED - CHECK MARGIN *
 7076   1A49   .   .   .   ;***************************************
 7077   1A49   .   .   .   CHD120 EQU  $
 7078   1A49   3A  BE  FF         LDA  RHTMGN
 7079   1A4C   B9  .   .          CMP  C         ;CHAR FROM BEYOND MARGIN?
 7080   1A4D   C2  57  1A         JNZ  CHD130    ;NO - CONTINUE DELETE
 7081   1A50   3A  98  FF         LDA  CHSAV     ;YES - GET DELETED CHARACTER
 7082   1A53   B7  .   .          ORA  A         ;IS IT ASCII?
 7083   1A54   36  20  .          MVI  M,ABLNK     ;(SET BLANK BY DEFAULT)
 7084   1A56   F0  .   .          RP             ;YES - TERMINATE DELETE
 7085   1A57   .   .   .   ;                     NO - PUT CHAR INTO PREV CHAR
 7086   1A57   .   .   .   ;***********************************************
 7087   1A57   .   .   .   ; MOVE CHARACTER INTO PREVIOUS CHARACTER POSITON *
 7088   1A57   .   .   .   ;***********************************************
 7089   1A57   .   .   .   CHD130 EQU  $
 7090   1A57   70  .   .          MOV  M,B       ;REPLACE PREVIOUS CHARACTER
 7091   1A58   78  .   .          MOV  A,B
 7092   1A59   B/  .   .          ORA  A         ;IS CHARACTER ASCII?
 7093   1A5A   FA  5E  1A         JM   CHD140    ;NO - ADVANCE TO NEXT CHAR
 7094   1A5D   0C  .   .          INR  C         ;YES - INCREMENT COLUMN #
 7095   1A5E   .   .   .   CHD140 EQU  $
 7096   1A5E   CD  86  0B         CALL NXTCH0    ;GET THE NEXT CHARACTER
 7097   1A61   FE  C5  .          CPI  ALPHA     ;TYPE DEFINITION?
 7098   1A63   DA  6C  1A         JC   CHD150    ;NO - CONTINUE MOVING CHARS
 7099   1A66   CD  91  1A         CALL CHD400    ;FORMAT MODE & DELETE ASCII?
 7100   1A69   C4  87  0B         CNZ  NXTCHR    ;YES - ADVANCE TO NEXT CHAR
 7101   1A6C   .   .   .   CHD150 EQU  $
 7102   1A6C   EB  .   .          XCHG           ;RESTORE REGISTER POSITIONS
 7103   1A6D   C3  22  1A         JMP  CHD110    ;MOVE NEXT CHARACTER
 7105    1A70   .    .    .   ;**********************************************
 7106    1A70   .    .    .   ; END OF LINE FILL CHARACTER FOUND - CLEAR THE *
 7107    1A70   .    .    .   ;   REST OF THE LINE                          *
 7108    1A70   .    .    .   ;**********************************************
 7109    1A70   .    .    .   CHD200 EQU  $
 7110    1A70   CD   68   19         CALL CHAINO    ;GET END OF LINE LINK IN H,L
 7111    1A73   EB   .    .          XCHG           ;EXCHANGE H,L AND D,E
 7112    1A74   .    .    .   ;*********************************************
 7113    1A74   .    .    .   ; END OF LINE LINK FOUND - CLEAR THE REST OF *
 7114    1A74   .    .    .   ;   THE LINE                                 *
 7115    1A74   .    .    .   ;*********************************************
 7116    1A74   .    .    .   CHD210 EQU  $
 7117    1A74   CD   91   1A         CALL CHD400    ;FORMAT MODE & DELETE ASCII?
 7118    1A77   EB   .    .          XCHG             ;(SET D,E TO LAST CHAR ADD
 7119    1A78   2B   .    .          DCX  H            ;H,L TO LSB OF NEXT LINE
 7120    1A79   .    .    .   ;                          LINK)
 7121    1A79   C2   8C   1A         JNZ  CHD260    ;YES - CLEAR REST OF FIELD
 7122    1A7C   .    .    .   ;                        TO LSB OF NEXT LINE LINK
 7123    1A7C   3A   98   FF         LDA  CHSAV     ;RECALL DELETED CHARACTER
 7124    1A7F   B7   .    .          ORA  A         ;WAS IT ASCII?
 7125    1A80   F2   8C   1A         JP   CHD260    ;YES - END LINE WITH EOL
 7126    1A83   3E   C3   .          MVI  A,FILL    ;NO - END LINE WITH FILL
 7127    1A85   C3   61   1C         JMP  CLERL1    ;CLEAR REST OF LINE
```

```
7129   1A88    .    .    .   ;*****************************************
7130   1A88    .    .    .   ; EOL FOUND - CLEAR THE REST OF THE LINE *
7131   1A88    .    .    .   ;*****************************************
7132   1A88    .    .    .   CHD250 EQU  $
7133   1A88    CD   68   19         CALL CHAINO     ;GET EOL LINK IN H,L
7134   1A8B    2B   .    .          DCX  H          ;SET TO LSB OF NEXT LINE LIN
7135   1A8C    .    .    .   CHD260 EQU  $          ;CLEAR THE REST OF THE LINE
7136   1A8C    3E   CC   .          MVI  A,EOL        ;TERMINATING WITH AN EOL
7137   1A8E    C3   61   1C         JMP  CLERL1
7138   1A91    .    .    .   ;********************************************
7139   1A91    .    .    .   ; CHD400 - CHECK FOR FORMAT MODE ENABLED AND *
7140   1A91    .    .    .   ;   DISPLAYABLE ASCII CHARACTER DELETED      *
7141   1A91    .    .    .   ;********************************************
7142   1A91    .    .    .   ;
7143   1A91    .    .    .   ;  EXIT :  NZ - FORMAT MODE AND DELETE ASCII
7144   1A91    .    .    .   ;          Z - NOT FORMAT MODE OR NON-DISPLAY
7145   1A91    .    .    .   ;            CODE DELETED
7146   1A91    .    .    .   ;
7147   1A91    .    .    .   CHD400 EQU  $
7148   1A91    3A   98   FF         LDA  CHSAV      ;GET CHARACTER DELETED
7149   1A94    B7   .    .          ORA  A          ;IS IT DISPAYABLE ASCII
7150   1A95    F2   7B   19         JP   CHKFMT     ;YES - CHECK FOR FORMAT MODE
7151   1A98    AF   .    .          XRA  A          ;NO - RETURN Z
7152   1A99    C9   .    .          RET
```

```
===========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                      PAGE 214
===========================================================================
 7154   1A9A    .   .   .   ;**********************************************
 7155   1A9A    .   .   .   ; CHD500 - CHECK FOR DISPLAY EHANCEMENT DELETE *
 7156   1A9A    .   .   .   ;**********************************************
 7157   1A9A    .   .   .   ;
 7158   1A9A    .   .   .   ;  ENTRY:  D,E = CHARACTER TO BE DELETED
 7159   1A9A    .   .   .   ;
 7160   1A9A    .   .   .   ;  EXIT :  D,E = ACTUAL CHARACTER TO DELETE
 7161   1A9A    .   .   .   ;               A,L DESTROYED
 7162   1A9A    .   .   .   ;
 7163   1A9A    .   .   .   CHD500 EQU  $
 7164   1A9A   1A   .   .          LDAX D          ;GET CHARACTER TO BE DELETED
 7165   1A9B   87   .   .          ADD  A          ;DISPLAY ENHANCEMENT CODE?
 7166   1A9C   D0   .   .          RNC             ;ASCII - LET IT BE DELETED
 7167   1A9D   F8   .   .          RM              ;FORMAT CONTROL - DELETE IT
 7168   1A9E   2E  02   .          MVI  L,2        ;YES - LOOK FOR POSSIBLE
 7169   1AA0   D5   .   .          PUSH D            ;DOUBLE EHANCEMENT CODE
 7170   1AA1    .   .   .   CHD510 EQU  $
 7171   1AA1   CD  87  0B          CALL NXICHR     ;GET THE NEXT CHARACTER
 7172   1AA4   C2  AE  1A          JNZ  CHD515     ;EXIT IF EOL LINK
 7173   1AA7   87   .   .          ADD  A          ;ENHANCEMENT CODE?
 7174   1AA8   D2  B0  1A          JNC  CHD520     ;ASCII - CHECK FOR SCAN DONE
 7175   1AAB   FA  A1  1A          JM   CHD510     ;FORMAT CONTROL - CONTINUE
 7176   1AAE    .   .   .   CHD515 EQU  $
 7177   1AAE   D1   .   .          POP  D          ;YES - DELETE ENHANCEMENT
 7178   1AAF   C9   .   .          RET
 7179   1AB0    .   .   .   ;*********************************************
 7180   1AB0    .   .   .   ; ASCII CHARACTER FOUND - CHECK FOR SCAN DONE *
 7181   1AB0    .   .   .   ;*********************************************
 7182   1AB0    .   .   .   CHD520 EQU  $
 7183   1AB0   2D   .   .          DCR  L          ;NEXT ASCII CHARACTER FOUND?
 7184   1AB1   C2  A1  1A          JNZ  CHD510     ;NO - CONTINUE SCAN
 7185   1AB4    .   .   .   ;*******************************************
 7186   1AB4    .   .   .   ; NEXT ASCII CHARACTER OR EOL LINK FOUND - *
 7187   1AB4    .   .   .   ;   DON'T DELETE DISPLAY ENHANCEMENT CODE  *
 7188   1AB4    .   .   .   ;*******************************************
 7189   1AB4   D1   .   .          POP  D          ;RECALL ORIGINAL ADDRESS
 7190   1AB5   C3  87  0B          JMP  NXTCHR     ;SET TO DELETE NEXT CHAR
```

```
7192   1AB8    .    .    .   ;*******************************************
7193   1AB8    .    .    .   ; CHRDL2 - DELETE CHARACTER w/REGISTER SAVE *
7194   1AB8    .    .    .   ;*******************************************
7195   1AB8    .    .    .   ;
7196   1AB8    .    .    .   ;  ENTRY:  C = CHARACTER COLUMN POSITION
7197   1AB8    .    .    .   ;          D,E = ADDRESS OF CHAR TO BE DELETED
7198   1AB8    .    .    .   ;
7199   1AB8    .    .    .   ;  EXIT :  B,C = B,C(ENTRY)
7200   1AB8    .    .    .   ;          D,E = D,E(ENTRY) + 1
7201   1AB8    .    .    .   ;          A,H,L DESTROYED
7202   1AB8    .    .    .   ;
7203   1AB8    .    .    .   CHRDL2 EQU  $
7204   1AB8    C5   .    .          PUSH B          ;SAVE REGISTERS B,C
7205   1AB9    D5   .    .          PUSH D            ;AND D,E
7206   1ABA    CD   19   1A         CALL CHRDL1     ;DELETE THE CHARACTER
7207   1ABD    D1   .    .          POP  D          ;RESTORE REGISTER D,E
7208   1ABE    C1   .    .          POP  B            ;AND B,C
7209   1ABF    13   .    .          INX  D          ;INCREMENT D,E
7210   1AC0    C9   .    .          RET             ;RETURN
```

```
 7212   1AC1    .   .   .   ;****************************
 7213   1AC1    .   .   .   ; CHRINS - INSERT CHARACTER *
 7214   1AC1    .   .   .   ;****************************
 7215   1AC1    .   .   .   ;
 7216   1AC1    .   .   .   ;  ENTRY:  A = CHARACTER TO BE INSERTED  .
 7217   1AC1    .   .   .   ;          CURROW,CURCOL = DISPLAY POSITION WHERE
 7218   1AC1    .   .   .   ;            INSERT IS TO BE DONE
 7219   1AC1    .   .   .   ;
 7220   1AC1    .   .   .   ;  EXIT :  A = 0, INSERT NOT DONE
 7221   1AC1    .   .   .   ;          A # 0, INSERT PERFORMED
 7222   1AC1    .   .   .   ;          DCHAR DESTROYED
 7223   1AC1    .   .   .   ;          B-L DESTROYED
 7224   1AC1    .   .   .   ;
 7225   1AC1    .   .   .   ;  CHARACTER IS INSERTED IMMEDIATELY AHEAD OF THE
 7226   1AC1    .   .   .   ;  CHARACTERS LOCATED AT THE SPECIFIED ROW AND
 7227   1AC1    .   .   .   ;  COLUMN POSITIONS
 7228   1AC1    .   .   .   ;
 7229   1AC1    .   .   .   CHRINS EQU  $
 7230   1AC1    32  89  FF         STA  DCHAR      ;SAVE CHAR TO BE INSERTED
 7231   1AC4    3E  FF  .          MVI  A,377Q     ;INHIBIT LINE EXTENSION
 7232   1AC6    32  91  FF         STA  BLKFIL
 7233   1AC9    CD  CD  06         CALL RCADR4     ;DOES DISPLAY POSITION EXIST
 7234   1ACC    C2  E9  21         JNZ  DISPLA     ;NO - TRY TO EXTEND LINE
 7235   1ACF    .   .   .   ;                       YES - INSERT THE CHARACTER
```

```
=============================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                          PAGE 217
=============================================================================
7237    1ACF   .   .   .   ;********************************************
7238    1ACF   .   .   .   ; CRI100 - ENTRY TO STORE CHARACTER FOR INSERT *
7239    1ACF   .   .   .   ;   CHARACTER MODE                             *
7240    1ACF   .   .   .   ;********************************************
7241    1ACF   .   .   .   ;
7242    1ACF   .   .   .   ;  ENTRY:  C = COLUMN NUMBER
7243    1ACF   .   .   .   ;          D,E = ADDR WHERE INSERT IS TO BE MADE
7244    1ACF   .   .   .   ;          H = BASEH
7245    1ACF   .   .   .   ;
7246    1ACF   .   .   .   CRI100 EQU  $
7247    1ACF   CD  65  10         CALL CKPROT    ;CURSOR IN PROTECTED FIELD?
7248    1AD2   CA  D4  22         JZ   DIS092    ;YES - TAB TO NEXT FIELD
7249    1AD5   .   .   .   CRI104 EQU $
7250    1AD5   2E  89  .          MVI  L,DCHAR-BASE
7251    1AD7   46  .   .          MOV  B,M       ;GET CHAR TO BE INSERTED
7252    1AD8   EB  .   .          XCHG           ;PUT CHAR ADDRESS INTO H,L
7253    1AD9   .   .   .   CRI110 EQU $
7254    1AD9   78  .   .          MOV A,B      ;IS THIS CONTROL CODE?
7255    1ADA   B7  .   .          ORA A
7256    1ADB   FA  DF  1A         JM   CRI120    ;YES - DON'T COUNT COLUMN
7257    1ADE   0C  .   .          INR C        ;INCREMENT COLUMN
7258    1ADF   .   .   .   CRI120 EQU $
7259    1ADF   7E  .   .          MOV A,M      ;GET CHAR IN CURRENT ADDR
7260    1AE0   70  .   .          MOV M,B      ;STORE NEW CHAR
7261    1AE1   47  .   .          MOV B,A      ;SAVE OLD CHAR IN B
7262    1AE2   2B  .   .          DCX  H         ;MOVE TO NEXT CHARACTER
7263    1AE3   3E  50  .          MVI  A,MAXCOL+1
7264    1AE5   B9  .   .          CMP  C         ;STORE DONE AT END OF LINE?
7265    1AE6   CA  91  1B         JZ   CRI305    ;YES - TERMINATE INSERT
7266    1AE9   3A  BE  FF         LDA  RHTMGN    ;GET RIGHT MARGIN COLUMN
7267    1AEC   3C  .   .          INR  A         ;WAS THE LAST STORE DONE
7268    1AED   B9  .   .          CMP  C           ;AT THE RIGHT MARGIN?
7269    1AEE   CA  89  1B         JZ   CRI300    ;YES - TERMINATE INSERT
```

```
 7271   1AF1    .    .    .    ;**********************************
 7272   1AF1    .    .    .    ; PROCESS NEXT CHARACTER OF BLOCK *
 7273   1AF1    .    .    .    ;**********************************
 7274   1AF1    .    .    .    CRI140 EQU  $
 7275   1AF1   7E    .    .           MOV  A,M        ;GET THE NEXT CHARACTER
 7276   1AF2   FE   C0    .           CPI  ENHLIM+1  ;ASCII OR DISPLAY CONTROL?
 7277   1AF4   DA   D9   1A           JC   CRI110     ;YES - MOVE THE BYTE
 7278   1AF7   FE   D0    .           CPI  LNKLIM     ;IS IT A LINK BYTE?
 7279   1AF9   D2   5D   1B           JNC CRI200    ;YES - MOVE TO NEXT BLOCK
 7280   1AFC   FE   CC    .           CPI EOL       ;IS IT END OF LINE?
 7281   1AFE   CA   39   1B           JZ   CRI158     ;YES - ADD LAST CHAR TO LINE
 7282   1B01   FE   C3    .           CPI  FILL       ;END OF LINE FILL CHARACTER?
 7283   1B03   CA   44   1B           JZ   CRI159     ;YES - ADD BYTE TO END
 7284   1B06   3A   89   FF           LDA  DCHAR      ;NO - FIELD CHECK CHARACTER
 7285   1B09   B7    .    .           ORA  A          ;IS ADDED CHARACTER ASCII?
 7286   1B0A   FA   D9   1A           JM   CRI110     ;NO - CONTINUE INSERT MOVE
 7287   1B0D   CD   76   19           CALL CHKFMS     ;FORMAT MODE ENABLED?
 7288   1B10   CA   D9   1A           JZ   CRI110     ;NO - CONTINUE INSERT
 7289   1B13   7E    .    .           MOV  A,M        ;YES - RECALL THE BYTE
 7290   1B14   FE   C0    .           CPI  STPR       ;IS CHARACTER A START PROT?
 7291   1B16   CA   22   1B           JZ   CRI150     ;YES - CHECK INSERT TYPE
 7292   1B19   FE   C5    .           CPI  ALPHA      ;FIELD TYPE DEFINITION?
 7293   1B1B   FA   D9   1A           JM   CRI110     ;NO - CONTINUE INSERT
 7294   1B1E   2B    .    .           DCX  H          ;YES - ADVANCE TO NEXT BYTE
 7295   1B1F   C3   F1   1A           JMP  CRI140     ;LOOK TO NEXT CHARACTER
```

```
7297   1B22    .    .    .    ;***********************************
7298   1B22    .    .    .    ; END OF FIELD - ASCII CODE INSERTED *
7299   1B22    .    .    .    ;***********************************
7300   1B22    .    .    .    CRI150 EQU  $
7301   1B22    78   .    .           MOV  A,B        ;GET CHAR WHICH ROLLED OFF
7302   1B23    B7   .    .           ORA  A          ;IS IT ASCII?
7303   1B24    F2   2F   1B          JP   CRI154     ;YES - DELETE PREV CONTROLS
7304   1B27    .    .    .    CRI152 EQU  $          ;NO - BACK UP ANOTHER CHAR
7305   1B27    CD   2E   1C          CALL CRI500     ;IS PREVIOUS CHARACTER ASCII
7306   1B2A    FA   27   1B          JM   CRI152     ;NO - CONTINUE BACKING UP
7307   1B2D    36   80   .           MVI  M,200Q     ;YES - TEMPORARILY REPLACE
7308   1B2F    .    .    .    ;                        ASCII WITH DUMMY CONTROL
7309   1B2F    .    .    .    CRI154 EQU  $
7310   1B2F    CD   2E   1C          CALL CRI500     ;PREVIOUS CHARACTER ASCII?
7311   1B32    FA   2F   1B          JM   CRI154     ;NO - CONTINUE BACKING UP
7312   1B35    2B   .    .           DCX  H          ;MOVE TO NEXT CHARACTER
7313   1B36    C3   9D   1C          JMP  CLER02     ;CLEAR REST OF FIELD
```

```
 7315   1B39    .    .    .   ;*****************************
 7316   1B39    .    .    .   ; EOL FOUND                   *
 7317   1B39    .    .    .   ; ADD LAST CHARACTER TO LINE *
 7318   1B39    .    .    .   ;*****************************
 7319   1B39    .    .    .   CRI158 EQU $
 7320   1B39    78   .    .          MOV A,B      ;GET CHARACTER
 7321   1B3A    B7   .    .          ORA A        ;IS THIS CONTROL CHAR?
 7322   1B3B    FA   47   1B         JM  CRI160   ;YES - ADD CHAR
 7323   1B3E    79   .    .          MOV A,C      ;NO - CHAR IS ASCII
 7324   1B3F    FE   4F   .          CPI MAXCOL   ;IS THIS MAX COLUMN?
 7325   1B41    C2   48   1B         JNZ CRI170   ;NO - ADD CHAR
 7326   1B44    .    .    .   CRI159 EQU $
 7327   1B44    70   .    .          MOV  M,B        ;ASCII CHARACTER INSERTED TO
 7328   1B45    B7   .    .          ORA  A            ;MAXIMUM COLUMN - OVERLAY
 7329   1B46    C9   .    .          RET               ;EOL AND RETURN NZ
 7330   1B47    .    .    .   ;**************************
 7331   1B47    .    .    .   ; EOL CANNOT BE OVERLAYED *
 7332   1B47    .    .    .   ; ADD NEW CHAR TO LINE    *
 7333   1B47    .    .    .   ;**************************
 7334   1B47    .    .    .   CRI160 EQU $
 7335   1B47    0D   .    .          DCR C
 7336   1B48    .    .    .   CRI170 EQU $
 7337   1B48    EB   .    .          XCHG            ;PUT H,L INTO D,E
 7338   1B49    .    .    .   CRI180 EQU $
 7339   1B49    21   89   FF         LXI  H,DCHAR    ;SAVE CHARACTER TO BE ADDED
 7340   1B4C    70   .    .          MOV  M,B
 7341   1B4D    2E   C1   .          MVI  L,CURCOL-BASE
 7342   1B4F    46   .    .          MOV  B,M        ;GET CURRENT CURSOR COLUMN
 7343   1B50    C5   .    .          PUSH B            ;AND SAVE IT
 7344   1B51    71   .    .          MOV  M,C        ;SET "CURCOL" TO INSERT COL
 7345   1B52    0E   00   .          MVI  C,0        ;SET # OF CHARS NEEDED TO 1
 7346   1B54    .    .    .   ;                        (VALUE IN C IS ONE LESS)
 7347   1B54    CD   AB   08         CALL DISPL1     ;BUILD NECESSARY BLOCKS
 7348   1B57    C1   .    .          POP  B          ;RESTORE ORIGINAL CURSOR
 7349   1B58    21   C1   FF         LXI  H,CURCOL     ;COLUMN NUMBER
 7350   1B5B    70   .    .          MOV  M,B
 7351   1B5C    C9   .    .          RET          ;RETURN (A=MEMORY LOCK STATE)
```

```
=========================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                   PAGE 221
=========================================================================
 7353   1B5D    .   .   .   ;***********************************
 7354   1B5D    .   .   .   ; LINK FOUND - MOVE TO NEXT BLOCK *
 7355   1B5D    .   .   .   ;***********************************
 7356   1B5D    .   .   .   CR1200 EQU   $
 7357   1B5D    22  96  FF         SHLD LNKSAV     ;SAVE CURRENT BLOCK ADDRESS
 7358   1B60    2B  .   .          DCX  H          ;GET THE LSB OF THE LINK
 7359   1B61    7E  .   .          MOV  A,M
 7360   1B62    2F  .   .          CMA             ;IS IT AN EOL LINK (LOWER
 7361   1B63    E6  0F  .          ANI  BLKSM        ;FOUR BITS NOT ALL ONES)?
 7362   1B65    C2  6E  1B         JNZ  CR1240     ;YES - EXTEND THE LINE
 7363   1B68    CD  6D  19         CALL CHAIN      ;NO - GET NEXT BLOCK ADDRESS
 7364   1B6B    C3  F1  1A         JMP  CR1140     ;CONTINUE INSERT CHARACTER
 7365   1B6E    .   .   .   ;*********************
 7366   1B6E    .   .   .   ; NEW BLOCK REQUIRED *
 7367   1B6E    .   .   .   ;*********************
 7368   1B6E    .   .   .   CR1240 EQU $
 7369   1B6E    78  .   .          MOV  A,B        ;SAVE CHARACTER BEING MOVED
 7370   1B6F    32  9D  FF         STA  TEMP
 7371   1B72    23  .   .          INX  H          ;GET THE LAST CHARACTER OF
 7372   1B73    23  .   .          INX  H            ;THE CURRENT BLOCK TO BE
 7373   1B74    46  .   .          MOV  B,M          ;STORED AGAIN IN THE SAME
 7374   1B75    EB  .   .          XCHG              ;LOCATION BY "DISPL1"
 7375   1B76    0D  .   .          DCR  C          ;GET COLUMN # OF PREV CHAR
 7376   1B77    CD  49  1B         CALL CR1180  ;ADD BLOCK
 7377   1B7A    B7  .   .          ORA A        ;IS MEMORY LOCKED?
 7378   1B7B    CA  83  1B         JZ  CR1260   ;YES - BLOCK NOT ADDED
 7379   1B7E    3A  9D  FF         LDA  TEMP       ;NO - RECALL CHAR TO BE ADDE
 7380   1B81    12  .   .          STAX D          ;PUT CHARACTER IN NEW BLOCK
 7381   1B82    .   .   .   ;                        (OVERWRITE EOL)
 7382   1B82    C9  .   .          RET             ;RETURN
 7383   1B83    .   .   .   ;*********************************
 7384   1B83    .   .   .   ; BLOCK NOT AVAILABLE           *
 7385   1B83    .   .   .   ; WRITE EOL AT END OF LAST BLOCK *
 7386   1B83    .   .   .   ;*********************************
 7387   1B83    .   .   .   CR1260 EQU $
 7388   1B83    2A  94  FF         LHLD EOLADR  ;GET ADR OF CHR BEFORE LNK
 7389   1B86    36  CC  .          MVI M,EOL    ;WRITE EOL
 7390   1B88    C9  .   .          RET          ;RETURN
```

```
==========================================================================
 ITEM     LOC    OBJECT CODE   SOURCE STATEMENTS
==========================================================================
 7392     1B89    .    .    .    ;*******************************************
 7393     1B89    .    .    .    ; RIGHT MARGIN OR END OF LINE REACHED -     *
 7394     1B89    .    .    .    ; TERMINATE AND OPTIONALLY PUSH CHARACTERS *
 7395     1B89    .    .    .    ; TO THE NEXT LINE (WRAP AROUND)           *
 7396     1B89    .    .    .    ;*******************************************
 7397     1B89    .    .    .    CRI300 EQU   $
 7398     1B89    3A   89   FF          LDA   DCHAR        ;GET THE INSERTED CHARACTER
 7399     1B8C    B7   .    .           ORA   A            ;IS IT ASCII?
 7400     1B8D    FA   F1   1A          JM    CRI140       ;NO - CONTINUE INSERTING
 7401     1B90    79   .    .           MOV   A,C          ;YES - RECALL ENDING COLUMN
 7402     1B91    .    .    .    CRI305 EQU   $
 7403     1B91    32   D7   FF          STA   PARM5        ;SAVE ENDING COLUMN NUMBER
 7404     1B94    22   96   FF          SHLD  LNKSAV       ;SAVE ENDING CHARACTER ADDR
 7405     1B97    EB   .    .           XCHG               ;PUT ENDING ADDRESS IN D,E
 7406     1B98    CD   05   26          CALL  INITD1       ;INIT CHAR BUFFER POINTERS
 7407     1B9B    13   .    .           INX   D            ;GET ADDRESS OF NEXT EXCESS
 7408     1B9C    CD   87   0B          CALL  NXTCHR         ;CHARACTER
 7409     1B9F    EB   .    .           XCHG
 7410     1BA0    22   D5   FF          SHLD  PARM6        ;ARE WE AT AN EOL LINK?
 7411     1BA3    78   .    .           MOV   A,B            ;(PUT 1ST EXCESS CHAR IN A
 7412     1BA4    CA   C3   1B          JZ    CRI320       ;NO - ACCUMULATE EXCESS
 7413     1BA7    CD   C0   13          CALL  A2OUTB       ;YES - SAVE FIRST EXCESS CHA
 7414     1BAA    B7   .    .           ORA   A            ;IS IT ASCII?
 7415     1BAB    F2   D8   1B          JP    CRI330       ;YES - CHECK FOR INSERT WRAP
 7416     1BAE    C9   .    .           RET                ;NO - RETURN
 7417     1BAF    .    .    .    ;
 7418     1BAF    .    .    .    ; ACCUMULATE THE EXCESS CHARACTERS
 7419     1BAF    .    .    .    ;
 7420     1BAF    .    .    .    CRI310 EQU   $
 7421     1BAF    2A   D5   FF          LHLD  PARM6        ;RECALL EXCESS CHAR ADDRESS
 7422     1BB2    EB   .    .           XCHG               ;PUT ADDRESS INTO D,E
 7423     1BB3    3E   FF   .           MVI   A,-1         ;SET DELETED CHAR TO -1
 7424     1BB5    32   98   FF          STA   CHSAV
 7425     1BB8    0E   50   .           MVI   C,MAXCOL+1  ;FORCE DELETE PAST MARGIN
 7426     1BBA    CD   19   1A          CALL  CHRDL1       ;DELETE ONE EXCESS CHARACTER
 7427     1BBD    3A   98   FF          LDA   CHSAV        ;RECALL THE DELETED CHARACTE
 7428     1BC0    47   .    .           MOV   B,A          ;SAVE THE CHARACTER IN B-REG
 7429     1BC1    04   .    .           INR   B            ;ANY CHARACTER DELETED?
 7430     1BC2    C8   .    .           RZ                 ;NO - RETURN (A#0)
 7431     1BC3    .    .    .    CRI320 EQU   $            ;YES - ACCUMULATE EXCESS
 7432     1BC3    CD   C0   13          CALL  A2OUTB       ;PUT DELETED CHAR INTO BUFFE
 7433     1BC6    B7   .    .           ORA   A            ;WAS DELETED CHARACTER ASCII
 7434     1BC7    FA   AF   1B          JM    CRI310       ;NO - CONTINUE ACCUMULATING
 7435     1BCA    3A   D7   FF          LDA   PARM5        ;RECALL ENDING COLUMN NUMBER
 7436     1BCD    FE   50   .           CPI   MAXCOL+1     ;TERMINATE ON LAST COLUMN?
 7437     1BCF    2A   96   FF          LHLD  LNKSAV         ;(RECALL ENDING CHAR ADDR)
 7438     1BD2    EB   .    .           XCHG
 7439     1BD3    3E   C3   .           MVI   A,FILL         ;(SET FOR FILL PAD)
 7440     1BD5    CC   5E   1C          CZ    CLERLO       ;YES - CLEAR REST OF LINE
```

```
7442   1BD8   .   .   .   ;
7443   1BD8   .   .   .   ; EXCESS CHARACTERS ACCUMULATED - CHECK FOR WRAP
7444   1BD8   .   .   .   ;
7445   1BD8   .   .   .   CRI330 EQU  $
7446   1BD8   3A  F8  FF         LDA  CMFLGS    ;GET THE COMMON FLAGS
7447   1BDB   2F  .   .          CMA            ;COMPLEMENT FLAGS
7448   1BDC   E6  02  .          ANI  INSWRP    ;WRAP AROUND ENABLED?
7449   1BDE   C0  .   .          RNZ            ;NO - RETURN (A#0)
7450   1BDF   3A  C1  FF         LDA  CURCOL    ;YES - GET THE CURRENT COLUM
7451   1BE2   47  .   .          MOV  B,A       ;SAVE VALUE IN B-REGISTER
7452   1BE3   3A  BE  FF         LDA  RHTMGN
7453   1BE6   B8  .   .          CMP  B         ;CURSOR BEYOND RIGHT MARGIN?
7454   1BE7   D8  .   .          RC             ;YES - RETURN
7455   1BE8   CD  76  19         CALL CHKFMS    ;FORMAT/SOFT KEY DEFINE MODE
7456   1BEB   C0  .   .          RNZ            ;YES - RETURN
7457   1BEC   C5  .   .          PUSH B         ;NO - SAVE CURRENT COLUMN AN
7458   1BED   21  C0  FF         LXI  H,CURROW  ;INCREMENT TO NEXT ROW
7459   1BF0   34  .   .          INR  M
7460   1BF1   2A  C9  FF         LHLD LSTLIN    ;CHECK TO SEE IF NEXT LINE
7461   1BF4   5E  .   .          MOV  E,M         ;IS FULL (I.E., NO "EOL"
7462   1BF5   23  .   .          INX  H           ;BEFORE RIGHT MARGIN)
7463   1BF6   56  .   .          MOV  D,M
7464   1BF7   1C  .   .          INR  E         ;DOES NEXT LINE EXIST?
7465   1BF8   1D  .   .          DCR  E           ;(LSB # 0)?
7466   1BF9   CA  0C  1C         JZ   CRI400    ;NO - ADD CHAR TO NEW LINE
7467   1BFC   13  .   .          INX  D         ;YES - START FROM BEGINNING
7468   1BFD   3A  BE  FF         LDA  RHTMGN      ;OF LINE TO RIGHT MARGIN
7469   1C00   CD  58  1F         CALL FNDLSO    ;NEXT LINE FULL?
7470   1C03   F2  0C  1C         JP   CRI400    ;NO - ADD OVERFLOW CHARACTER
7471   1C06   .   .   .    ;                      TO NEXT LINE
7472   1C06   CD  00  0A         CALL LININS    ;YES - INSERT A LINE
7473   1C09   CA  25  1C         JZ   CRI450    ;EXIT IF MEMORY LOCKED
```

```
 7475    1C0C    .    .    .    ;
 7476    1C0C    .    .    .    ;  INSERT CHARACTERS INTO NEXT LINE
 7477    1C0C    .    .    .    ;
 7478    1C0C    .    .    .    CRI400 EQU  $
 7479    1C0C    21   3B   FF          LXI  H,B2DEND  ;GET BUFFER POINTER
 7480    1C0F    7E   .    .           MOV  A,M
 7481    1C10    FE   3C   .           CPI  B2DBFL-1  ;ALL BYTES DONE?
 7482    1C12    CA   25   1C          JZ   CRI450    ;YES - EXIT
 7483    1C15    35   .    .           DCR  M         ;NO - UPDATE BUFFER POINTER
 7484    1C16    6F   .    .           MOV  L,A       ;PUT LSB INTO L
 7485    1C17    3A   BF   FF          LDA  LFTMGN    ;SET TO INSERT CHARACTER AT
 7486    1C1A    32   C1   FF          STA  CURCOL      ;LEFT MARGIN
 7487    1C1D    7E   .    .           MOV  A,M       ;GET CHARACTER TO INSERT
 7488    1C1E    CD   C1   1A          CALL CHRINS    ;INSERT CHARACTER
 7489    1C21    B7   .    .           ORA  A         ;INSERT SUCCESSFUL?
 7490    1C22    C2   0C   1C          JNZ  CRI400    ;YES - DO NEXT BYTE
 7491    1C25    .    .    .    ;
 7492    1C25    .    .    .    ;  ALL CHARACTERS INSERTED - EXIT
 7493    1C25    .    .    .    ;
 7494    1C25    .    .    .    CRI450 EQU  $
 7495    1C25    21   C0   FF          LXI  H,CURROW
 7496    1C28    35   .    .           DCR  M         ;RESTORE THE ROW NUMBER
 7497    1C29    F1   .    .           POP  PSW       ;RECALL THE COLUMN NUMBER
 7498    1C2A    23   .    .           INX  H
 7499    1C2B    77   .    .           MOV  M,A       ;RESTORE COLUMN NUMBER
 7500    1C2C    3C   .    .           INR  A         ;FORCE A # 0
 7501    1C2D    C9   .    .           RET            ;RETURN
```

```
==============================================================================
 ITEM     LOC   OBJECT CODE  SOURCE STATEMENTS                          PAGE 225
==============================================================================
 7503    1C2E    .   .   .   ;**********************************
 7504    1C2E    .   .   .   ; CR1500 - GET PREVIOUS CHARACTER *
 7505    1C2E    .   .   .   ;**********************************
 7506    1C2E    .   .   .   ;
 7507    1C2E    .   .   .   ;  ENTRY:  H,L = CURRENT CHARACTER ADDRESS
 7508    1C2E    .   .   .   ;          LNKSAV = ADDRESS OF MSB PART OF NEXT
 7509    1C2E    .   .   .   ;            BLOCK LINK IN PREVIOUS BLOCK
 7510    1C2E    .   .   .   ;
 7511    1C2E    .   .   .   ;  EXIT :  A = PREVIOUS CHARACTER
 7512    1C2E    .   .   .   ;          H,L = ADDRESS OF PREVIOUS CHARACTER
 7513    1C2E    .   .   .   ;          P - CHARACTER IS ASCII
 7514    1C2E    .   .   .   ;          M - CHARACTER IS NON-DISPLAY CONTROL
 7515    1C2E    .   .   .   ;
 7516    1C2E    .   .   .   CRI500 EQU   $
 7517    1C2E    23  .   .          INX   H           ;MOVE TO PREVIOUS CHARACTER
 7518    1C2F    7D  .   .          MOV   A,L           ;IN BLOCK
 7519    1C30    F6  0F  .          ANI   BLKSM       ;PREVIOUS CHARACTER IN BLOCK
 7520    1C32    C2  39  1C         JNZ   CRI510      ;YES - GET IT
 7521    1C35    2A  96  FF         LHLD  LNKSAV      ;NO - GET PREV BLOCK ADDRESS
 7522    1C38    23  .   .          INX   H           ;SET TO LAST CHARACTER ADDR
 7523    1C39    .   .   .   CRI510 EQU   $
 7524    1C39    7E  .   .          MOV   A,M         ;GET THE PREVIOUS CHARACTER
 7525    1C3A    B7  .   .          ORA   A           ;SET FLAGS FOR ASCII OR NOT
 7526    1C3B    C9  .   .          RET                 ;ASCII AND RETURN
 7528   1C3C    .    .    .   ;**********************
 7529   1C3C    .    .    .   ; CLEARL - CLEAR LINE *
 7530   1C3C    .    .    .   ;**********************
 7531   1C3C    .    .    .   ;
 7532   1C3C    .    .    .   ;  ENTRY:  DON'T CARE
 7533   1C3C    .    .    .   ;
 7534   1C3C    .    .    .   ;  EXIT :  A = -1, ROW NOT FOUND
 7535   1C3C    .    .    .   ;            =  0, CHARACTER FOUND AND CLEAR DONE
 7536   1C3C    .    .    .   ;            >  0, COLUMN PAST EOL, CLEAR NOT DONE
 7537   1C3C    .    .    .   ;
 7538   1C3C    .    .    .   CLEARL EQU  $
 7539   1C3C    CD   CD   06         CALL RCADR4     ;DOES ROW EXIST?
 7540   1C3F    C0   .    .          RNZ             ;NO - RETURN
 7541   1C40    CD   76   19         CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
 7542   1C43    CA   54   1C         JZ   CLERLA     ;NO - DO NORMAL CLEAR LINE
 7543   1C46    F2   9A   1C         JP   CLL400     ;FORMAT MODE - CLEAR FIELD
 7544   1C49    .    .    .   ;***********************************************
 7545   1C49    .    .    .   ; SOFT KEY DEFINE MODE - CLEAR DATA ROWS ONLY *
 7546   1C49    .    .    .   ;***********************************************
 7547   1C49    3A   C0   FF         LDA  CURROW     ;GET CURSOR ROW
 7548   1C4C    0F   .    .          RRC             ;IN DATA LINE (ODD ROW #)?
 7549   1C4D    D0   .    .          RNC             ;NO - INHIBIT CLEAR
 7550   1C4E    1A   .    .          LDAX D          ;GET FIRST CHARACTER
 7551   1C4F    FE   C1   .          CPI  ENDPR      ;END PROTECT?
 7552   1C51    CC   87   0B         CZ   NXTCHR     ;YES - SKIP TO 1ST ASCII CHA
 7553   1C54    .    .    .   CLERLA EQU  $
 7554   1C54    CD   8C   19         CALL CHKSFK     ;SOFT KEY DEFINE MODE?
 7555   1C57    3E   0C   .          MVI  A,SETFRN    ;(SET CONTROL CODE)
 7556   1C59    CC   08   48         CZ   ZKBCTL     ;NO - UPDATE FOREIGN MODE
 7557   1C5C    3E   CC   .          MVI  A,EOL      ;CLEAR LINE WITH "EOL" ENDIN
```

```
7559    1C5E    .   .   .    ;*******************************
7560    1C5E    .   .   .    ; CLERLO - CLEAR REST OF LINE *
7561    1C5E    .   .   .    ;*******************************
7562    1C5E    .   .   .    ;
7563    1C5E    .   .   .    ;   ENTRY:  A = TERMINATOR CHARACTER
7564    1C5E    .   .   .    ;           D,E = CLEAR STARTING ADDRESS
7565    1C5E    .   .   .    ;
7566    1C5E    .   .   .    ;   EXIT :  SEE "CLEARL"
7567    1C5E    .   .   .    ;
7568    1C5E    .   .   .    CLERLO EQU   $
7569    1C5E    2A  C9  FF          LHLD LSILIN      ;GET CURRENT LINE ADDRESS
7570    1C61    .   .   .    CLERL1 EQU   $
7571    1C61    32  8F  FF          STA   FILCHR     ;SAVE TERMINATOR CHARACTER
7572    1C64    44  .   .           MOV   B,H        ;SET B,C TO ADDRESS OF NEXT
7573    1C65    4D  .   .           MOV   C,L          ;LINE POINTER'S LSB
7574    1C66    7B  .   .           MOV   A,E        ;SET H,L TO ADDRESS OF NEXT
7575    1C67    E6  F0  .           ANI   377Q-BLKSM   ;BLOCK LINK IN CURRENT
7576    1C69    6F  .   .           MOV   L,A          ;BLOCK
7577    1C6A    62  .   .           MOV  H,D
7578    1C6B    7E  .   .           MOV  A,M       ;GET NEXT BLOCK
7579    1C6C    03  .   .           INX   B          ;SET B,C TO MSB OF NEXT LINE
7580    1C6D    71  .   .           MOV   M,C          ;POINTER
7581    1C6E    23  .   .           INX   H
7582    1C6F    4E  .   .           MOV   C,M
7583    1C70    70  .   .           MOV   M,B
7584    1C71    45  .   .           MOV   B,L        ;SAVE LSB OF LINK'S MSB ADDR
7585    1C72    61  .   .           MOV   H,C
7586    1C73    6F  .   .           MOV   L,A
7587    1C74    E5  .   .           PUSH H           ;SAVE ADDRESS OF NEXT BLOCK
```

```
7589   1C75   .  .  .    ;******************************************
7590   1C75   .  .  .    ; INSERT FILL CHARS BETWEEN LINK AND EOL *
7591   1C75   .  .  .    ;******************************************
7592   1C75   7B .  .              MOV A,E        ;COMPUTE NO. OF FILLS
7593   1C76   E6 0F .              ANI BLKSM
7594   1C78   D6 02 .              SUI 2          ;LESS THAN 2?
7595   1C7A   FA 8A 1C             JM  CLL160     ;YES - RELEASE THE BLOCK
7596   1C7D   58 .  .              MOV  E,B         ;SET H,L TO ADDRESS OF MSB
7597   1C7E   EB .  .              XCHG               ;PART OF NEXT BLOCK POINTE
7598   1C7F   .  .  .    CLL120 EQU $
7599   1C7F   23 .  .              INX  H           ;ADVANCE TO NEXT BYTE
7600   1C80   36 C3 .              MVI  M,FILL      ;STORE FILL CHARACTER
7601   1C82   3D .  .              DCR  A           ;ALL BYTES DONE?
7602   1C83   F2 7F 1C             JP   CLL120      ;NO - CONTINUE FILLING
7603   1C86   3A 8F FF             LDA  FILCHR      ;YES - GET AND STORE FINAL
7604   1C89   77 .  .              MOV  M,A           ;FILL CHARACTER
7605   1C8A   .  .  .    ;********************************
7606   1C8A   .  .  .    ; RELEASE EXCESS DISPLAY BLOCKS *
7607   1C8A   .  .  .    ;********************************
7608   1C8A   .  .  .    CLL160 EQU  $
7609   1C8A   D1 .  .              POP   D          ;RECALL ADDRESS OF NEXT BLOC
7610   1C8B   7B .  .              MOV   A,E
7611   1C8C   2F .  .              CMA              ;IS THE LINK AN EOL LINK
7612   1C8D   E6 0F .              ANI   BLKSM        ;(LOW 4 BITS NOT ALL ONES)
7613   1C8F   C2 98 1C             JNZ   CLL310     ;YES - EXIT
7614   1C92   1B .  .              DCX   D          ;NO - ADD BLOCKS TO FREE LIS
7615   1C93   1B .  .              DCX   D          ;SET ADDRESS TO LSB OF NEXT
7616   1C94   1B .  .              DCX   D            ;LINE FIELD IN FIRST BLOCK
7617   1C95   CD 91 06             CALL PUTLIN      ;ADD BLOCKS TO FREE LIST
7618   1C98   .  .  .    CLL310 EQU $
7619   1C98   AF .  .              XRA A          ;SET ZERO FLAG FOR CLEARS
7620   1C99   C9 .  .              RET            ;RETURN
```

```
===============================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                        PAGE 229
===============================================================================
 7622    1C9A   .   .   .   ;***************************************
 7623    1C9A   .   .   .   ; CLEAR LINE FUNCTION FOR FORMAT MODE *
 7624    1C9A   .   .   .   ;***************************************
 7625    1C9A   .   .   .   CLL400 EQU  $
 7626    1C9A   04  .   .          INR  B         ;CURSOR IN PROTECTED FIELD?
 7627    1C9B   C8  .   .          RZ             ;YES - RETURN, DON'T DO CLEA
 7628    1C9C   .   .   .   ;*********************************************
 7629    1C9C   .   .   .   ; CLEAR UNPROTECTED FIELD                    *
 7630    1C9C   .   .   .   ; D,E = ADDRESS OF FIRST ASCII CHAR IN FIELD *
 7631    1C9C   .   .   .   ;*********************************************
 7632    1C9C   .   .   .   CLER01 EQU $
 7633    1C9C   EB  .   .          XCHG
 7634    1C9D   .   .   .   CLER02 EQU  $
 7635    1C9D   EB  .   .          XCHG           ;PUT CHARACTER ADDR INTO D,E
 7636    1C9E   13  .   .          INX  D         ;SET TO PREVIOUS CHARACTER
 7637    1C9F   .   .   .   CLL510 EQU  $
 7638    1C9F   CD  87  0B         CALL NXTCHR    ;GET THE NEXT CHARACTER
 7639    1CA2   C2  CA  1C         JNZ  CLL580    ;CHECK EXIT IF EOL LINK
 7640    1CA5   87  .   .          ADD A        ;ASCII?
 7641    1CA6   DA  B0  1C         JC  CLL540   ;NO - CONTINUE
 7642    1CA9   3E  20  .          MVI  A,ABLNK   ;YES - STORE BLANK
 7643    1CAB   12  .   .          STAX D
 7644    1CAC   0C  .   .          INR C        ;INCREMENT COLUMN
 7645    1CAD   C3  9F  1C         JMP  CLL510    ;TRY NEXT CHARACTER
 7646    1CB0   .   .   .   ;**********************
 7647    1CB0   .   .   .   ; NON-ASCII CHARACTER *
 7648    1CB0   .   .   .   ;**********************
 7649    1CB0   .   .   .   CLL540 EQU $
 7650    1CB0   FA  B9  1C         JM  CLL550   ;NOT DSPLY CNTRL - CHECK MORE
 7651    1CB3   .   .   .   ;**********************************
 7652    1CB3   .   .   .   ; DELETE DISPLAY ENHANCEMENT CHAR *
 7653    1CB3   .   .   .   ;**********************************
 7654    1CB3   .   .   .   CLL544 EQU  $
 7655    1CB3   CD  B8  1A         CALL CHRDL2    ;DELETE ENHANCEMENT CODE
 7656    1CB6   C3  9F  1C         JMP  CLL510    ;CONTINUE CLEARING
```

```
=========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=========================================================================
 7658   1CB9   .   .   .   ;********************************
 7659   1CB9   .   .   .   ; NOT ASCII OR DISPLAY CONTROL *
 7660   1CB9   .   .   .   ;********************************     .
 7661   1CB9   .   .   .   CLL550 EQU  $
 7662   1CB9   1F  .   .          RAR            ;RESTORE CHARACTER
 7663   1CBA   FE  C3  .          CPI  FILL      ;END OF LINE FILL?
 7664   1CBC   CA  9F  1C         JZ   CLL510    ;YES - GO TO NEXT CHARACTER
 7665   1CBF   FE  C0  .          CPI  STPR      ;START PROTECT?
 7666   1CC1   C8  .   .          RZ             ;YES - TERMINATE CLEAR
 7667   1CC2   FE  C5  .          CPI  STPFLG+1  ;FORMAT CONTROL CODE?
 7668   1CC4   DA  B3  1C         JC   CLL544    ;YES - DELETE IT
 7669   1CC7   C3  9F  1C         JMP  CLL510    ;NO - GO TO NEXT CHARACTER
 7670   1CCA   .   .   .   ;*********************
 7671   1CCA   .   .   .   ; LINK FOUND         *
 7672   1CCA   .   .   .   ; MOVE TO NEXT BLOCK *
 7673   1CCA   .   .   .   ;*********************
 7674   1CCA   .   .   .   CLL580 EQU  $
 7675   1CCA   1A  .   .          LDAX D         ;GET NEXT LINE LINK'S MSB
 7676   1CCB   FE  CE  .          CPI  EOP       ;END OF DISPLAY LIST?
 7677   1CCD   C8  .   .          RZ             ;YES - RETURN
 7678   1CCE   CD  84  1E         CALL FLDSR2    ;CONTINUATION FIELD?
 7679   1CD1   CA  9F  1C         JZ   CLL510    ;YES - CONTINUE CLEAR
 7680   1CD4   AF  .   .          XRA  A         ;NO - TERMINATE CLEAR AND
 7681   1CD5   C9  .   .          RET              ;RETURN END ON END OF FIEL
```

```
7683    1CD6    .   .   .   ;***************************
7684    1CD6    .   .   .   ; DSPMSG - DISPLAY MESSAGE *
7685    1CD6    .   .   .   ;***************************
7686    1CD6    .   .   .   ;
7687    1CD6    .   .   .   ;   ENTRY:  NC - ADD MESSAGE TO NORMAL DISPLAY
7688    1CD6    .   .   .   ;           C - REPLACE DISPLAY WITH MESSAGE
7689    1CD6    .   .   .   ;           MSGPT1-MSGPT8 = POINTERS TO MESSAGE
7690    1CD6    .   .   .   ;             SECTIONS
7691    1CD6    .   .   .   ;
7692    1CD6    .   .   .   ;   EXIT :  ALL REGISTERS DESTROYED
7693    1CD6    .   .   .   ;
7694    1CD6    .   .   .   DSPMS0 EQU  $           ;SET C-FLAG TO FORCE DISPLAY
7695    1CD6    37  .   .          STC                ;REPLACEMENT BY MESSAGE
7696    1CD7    .   .   .   DSPMS1 EQU  $
7697    1CD7    22  F1  FF         SHLD MSGPT1      ;SET MESSAGE POINTER 1
7698    1CDA    .   .   .   DSPMSG EQU  $
7699    1CDA    D2  FB  1C         JNC  DSM500      ;ADD MESSAGE TO DISPLAY
7700    1CDD    01  4F  FE         LXI  B,DSPSTR    ;SET DESTINATION POINTER
7701    1CE0    21  F2  FF         LXI  H,MSGPT1+1  ;SET INITIAL TABLE POINTER
7702    1CE3    .   .   .   ;
7703    1CE3    .   .   .   ;   TRANSFER MESSAGE TO MESSAGE BUFFER
7704    1CE3    .   .   .   ;
7705    1CE3    .   .   .   DSM010 EQU  $
7706    1CE3    56  .   .          MOV  D,M         ;GET POINTER TO MESSAGE
7707    1CE4    2B  .   .          DCX  H
7708    1CE5    5E  .   .          MOV  E,M
7709    1CE6    2B  .   .          DCX  H           ;SET TO NEXT POINTER
7710    1CE7    EB  .   .          XCHG             ;PUT POINTER INTO H,L
7711    1CE8    CD  20  0B         CALL MOVCHR      ;XFR MESSAGE PART TO BUFFER
7712    1CEB    EB  .   .          XCHG             ;PUT POINTER TO TABLE IN H,L
7713    1CEC    CA  E3  1C         JZ   DSM010      ;DO NEXT PART IF NOT EOP END
7714    1CEF    21  4F  FE         LXI  H,DSPSTR    ;SET DISPLAY POINTER TO
7715    1CF2    22  FE  FF         SHLD DISPST        ;MESSAGE AREA
7716    1CF5    3E  18  .          MVI  A,MAXROW+1  ;REMOVE CURSOR FROM DISPLA
7717    1CF7    32  20  87         STA  IOCRRW
7718    1CFA    C9  .   .          RET              ;RETURN
```

```
 7720   1CFB   .   .   .    ;
 7721   1CFB   .   .   .    ;  ADD MESSAGE TO NORMAL DISPLAY
 7722   1CFB   .   .   .    ;
 7723   1CFB   .   .   .    DSM500 EQU  $
 7724   1CFB   CD  8D  0C          CALL SFKYOF     ;FORCE NORMAL DISPLAY ON
 7725   1CFE   21  F2  FF          LXI  H,MSGPT1+1  ;SET INITIAL TABLE POINTER
 7726   1D01   .   .   .    DSM510 EQU  $
 7727   1D01   56  .   .           MOV  D,M        ;GET POINTER TO MESSAGE
 7728   1D02   2B  .   .           DCX  H
 7729   1D03   5E  .   .           MOV  E,M
 7730   1D04   2B  .   .           DCX  H          ;SET TO NEXT POINTER
 7731   1D05   E5  .   .           PUSH H          ;SAVE TABLE POINTER
 7732   1D06   EB  .   .           XCHG            ;PUT MESSAGE POINTER IN H,L
 7733   1D07   CD  CD  0F          CALL XMS2DS     ;XFR MESSAGE TO THE DISPLAY
 7734   1D0A   E1  .   .           POP  H          ;RECALL TABLE POINTER
 7735   1D0B   CA  01  1D          JZ   DSM510     ;DO NEXT PART IF NOT EOP END
 7736   1D0E   .   .   .    ;                      FALL INTO "RSTDSP" TO
 7737   1D0E   .   .   .    ;                        FORCE DISPLAY ON
 7738   1D0E   .   .   .    ;**********************************
 7739   1D0E   .   .   .    ; RSTDSP - RESTORE NORMAL DISPLAY *
 7740   1D0E   .   .   .    ;**********************************
 7741   1D0E   .   .   .    ;
 7742   1D0E   .   .   .    ;  ENTRY:  DON'T CARE
 7743   1D0E   .   .   .    ;
 7744   1D0E   .   .   .    ;  EXIT :  PROCESSOR FLAGS UNCHANGED
 7745   1D0E   .   .   .    ;          H,L DESTROYED
 7746   1D0E   .   .   .    ;
 7747   1D0E   .   .   .    RSTDSP EQU  $
 7748   1D0E   2A  CB  FF          LHLD TOPLIN     ;GET TOP LINE ADDRESS
 7749   1D11   2B  .   .           DCX  H          ;SET TO FIRST CHAR ADDRESS
 7750   1D12   22  FE  FF          SHLD DISPST     ;SET DISPLAY START POINTER
 7751   1D15   C3  9E  0F          JMP  DISLN1     ;SET THE DISPLAY CURSOR
```

```
7753   1D18   .   .   .   ;*****************************
7754   1D18   .   .   .   ; FORMON - ENTER FORMAT MODE *
7755   1D18   .   .   .   ;*****************************
7756   1D18   .   .   .   FORMON EQU  $
7757   1D18   CD  4D  10         CALL CKEDIT     ;EDIT MODE?
7758   1D1B   C0  .   .          RNZ             ;YES - INHIBIT FORMAT MODE
7759   1D1C   21  4F  00         LXI  H,MAXCOL   ;NO - SET MARGINS TO ENDS OF
7760   1D1F   22  BE  FF         SHLD RHTMGN       ;DISPLAY
7761   1D22   3E  08  .          MVI  A,FORMAT   ;TURN ON FORMAT MODE FLAG
7762   1D24   CD  0E  48         CALL ZSTMD1
7763   1D27   .   .   .   ;                           SET CURSOR TO FIRST
7764   1D27   .   .   .   ;                             UNPROTECTED FIELD
7765   1D27   .   .   .   ;
7766   1D27   .   .   .   ;***********************************
7767   1D27   .   .   .   ; CURPH - CURSOR POINTER HOME (UP) *
7768   1D27   .   .   .   ;***********************************
7769   1D27   .   .   .   CURPH  EQU $
7770   1D27   3E  FE  .          MVI  A,377Q-SDACOM  ;CLEAR DATACOM INPUT
7771   1D29   CD  01  16         CALL CLRDFL       ;FLAG TO DISABLE TRANSMIT-
7772   1D2C   .   .   .   ;                        ONLY FIELDS
7773   1D2C   .   .   .   ;
7774   1D2C   .   .   .   CURPH1 EQU  $
7775   1D2C   CD  C5  21         CALL CURPRT     ;SET CURSOR TO LEFT MARGIN
7776   1D2F   CD  8C  19         CALL CHKSFK     ;SOFT KEY MODE?
7777   1D32   C2  92  1D         JNZ  HUP060     ;YES - SET CURSOR ONLY
7778   1D35   32  A3  FF         STA  TLINO      ;NO - SET TOP LINE # TO ZERO
7779   1D38   3D  .   .          DCR  A          ;RESET SPOW LATCH
7780   1D39   32  6C  FF         STA  SPOWL
7781   1D3C   CD  EE  0A         CALL MLKSCH     ;DISPLAY AREA LOCKED?
7782   1D3F   CA  A3  1D         JZ   HUP100     ;NO - HOME TO FIRST LINE
7783   1D42   .   .   .   ;
7784   1D42   .   .   .   ;  DISPLAY LOCK ON - CHANGE ONLY UNLOCKED LINES
7785   1D42   .   .   .   ;
7786   1D42   54  .   .          MOV  D,H        ;SAVE ADDRESS OF LSB PART OF
7787   1D43   5D  .   .          MOV  E,L          ;NEXT LINE POINTER IN FIRS
7788   1D44   23  .   .          INX  H            ;UNLOCKED LINE
7789   1D45   23  .   .          INX  H          ;GET ADDRESS OF LAST LOCKED
7790   1D46   4E  .   .          MOV  C,M          ;ROW
7791   1D47   23  .   .          INX  H
7792   1D48   46  .   .          MOV  B,M
7793   1D49   2A  CB  FF         LHLD TOPLIN     ;GET PTR TO TOP DSPLY LINE
7794   1D4C   23  .   .          INX  H          ;GET ADDRESS OF FIRST LINE
7795   1D4D   23  .   .          INX  H            ;ABOVE TOP DISPLAY LINE
7796   1D4E   7E  .   .          MOV  A,M
7797   1D4F   B7  .   .          ORA  A          ;ANY LINES ABOVE DISPLAY?
7798   1D50   CA  7D  1D         JZ   HUP050     ;NO - POSITION CURSOR ONLY
```

```
==========================================================================
 ITEM     LOC    OBJECT CODE  SOURCE STATEMENTS                  PAGE 234
==========================================================================
 7800    1D53    .     .     .    ;
 7801    1D53    .     .     .    ;  LINK SPLIT PARTS TOGETHER
 7802    1D53    .     .     .    ;
 7803    1D53    36    00    .          MVI  M,0         ;ZERO PREV LINE PTR OF TOP L
 7804    1D55    23    .     .          INX  H
 7805    1D56    66    .     .          MOV  H,M         ;SET H,L TO FIRST LINE ABOVE
 7806    1D57    6F    .     .          MOV  L,A           ;DISPLAY
 7807    1D58    1B    .     .          DCX  D           ;SET ITS NEXT LINE POINTER T
 7808    1D59    73    .     .          MOV  M,E           ;FIRST CHARACTER OF FIRST
 7809    1D5A    23    .     .          INX  H           ;UNLOCKED LINE
 7810    1D5B    72    .     .          MOV  M,D
 7811    1D5C    EB    .     .          XCHG             ;SET PREVIOUS LINE POINTER O
 7812    1D5D    23    .     .          INX  H             ;FIRST UNLOCKED LINE TO
 7813    1D5E    23    .     .          INX  H             ;FIRST LINE ABOVE DISPLAY
 7814    1D5F    23    .     .          INX  H
 7815    1D60    77    .     .          MOV  M,A
 7816    1D61    23    .     .          INX  H
 7817    1D62    72    .     .          MOV  M,D
 7818    1D63    2A    9F    FF         LHLD FLINE       ;REPLACE CONTENTS OF FLINE
 7819    1D66    EB    .     .          XCHG               ;WITH VALUES FROM TOPLIN
 7820    1D67    2A    CB    FF         LHLD TOPLIN
 7821    1D6A    22    9F    FF         SHLD FLINE
 7822    1D6D    62    .     .          MOV  H,D         ;SET PREVIOUS LINE POINTER O
 7823    1D6E    6B    .     .          MOV  L,E           ;CURRENT TOP LINE TO POINT
 7824    1D6F    23    .     .          INX  H             ;TO LAST LOCKED ROW
 7825    1D70    23    .     .          INX  H
 7826    1D71    71    .     .          MOV  M,C
 7827    1D72    23    .     .          INX  H
 7828    1D73    70    .     .          MOV  M,B
 7829    1D74    60    .     .          MOV  H,B         ;SET H,L TO MSB PART OF NEXT
 7830    1D75    69    .     .          MOV  L,C           ;LINE POINTER IN LAST
 7831    1D76    23    .     .          INX  H             ;LOCKED ROW
 7832    1D77    42    .     .          MOV  B,D         ;SET NEXT LINE POINTER TO
 7833    1D78    4B    .     .          MOV  C,E           ;POINT TO FIRST CHARACTER
 7834    1D79    0B    .     .          DCX  B             ;OF LINE POINTED BY FLINE
 7835    1D7A    CD    95    0F         CALL DISLNK
 7836    1D7D    .     .     .    ;
 7837    1D7D    .     .     .    ;  DISPLAY SET FOR DISPLAY LOCK HOME - SET CURSOR
 7838    1D7D    .     .     .    ;
 7839    1D7D    .     .     .    HUP050 EQU  $
 7840    1D7D    CD    56    0C         CALL ROLUP3      ;SET "LSTLIN" AND "CURADR"
 7841    1D80    CD    7B    19         CALL CHKFMT      ;FORAMT MODE?
 7842    1D83    EE    08    .          XRI  FORMAT        ;(REVERSE RESULT OF TEST)
 7843    1D85    CA    A6    1D         JZ   HUP110      ;YES - LOCATE FIRST FIELD
 7844    1D88    .     .     .    ;                        STARTING IN LOCKED REGION
 7845    1D88    .     .     .    ;                        (A = 0)
 7846    1D88    3A    6B    FF         LDA  MLKROW      ;NO - SET CURSOR TO FIRST
 7847    1D8B    32    C0    FF         STA  CURROW        ;UNLOCKED ROW
 7848    1D8E    32    C7    FF         STA  LSTROW
 7849    1D91    C9    .     .          RET              ;RETURN
 7851    1D92    .    .    .    ;
 7852    1D92    .    .    .    ;  DEFINE SOFT KEYS HOME UP
 7853    1D92    .    .    .    ;
 7854    1D92    .    .    .    HUP060 EQU   $
 7855    1D92    AF   .    .           XRA   A           ;SET CURSOR ROW TO ZERO
 7856    1D93    32   C0   FF          STA   CURROW
 7857    1D96    32   C7   FF          STA   LSTROW
 7858    1D99    2A   A6   FF          LHLD  SFTKYS      ;SET "CURADR" AND "LSTLIN"
 7859    1D9C    23   .    .           INX   H             ;TO FIRST SOFT KEY LINE
 7860    1D9D    CD   57   0C          CALL  ROLUPC
 7861    1DA0    C3   B9   1D          JMP   FLDSR1      ;LOCATE FIRST FIELD
 7862    1DA3    .    .    .    ;
 7863    1DA3    .    .    .    ;  DISPLAY NOT LOCKED - SET TOPLIN TO FLINE
 7864    1DA3    .    .    .    ;
 7865    1DA3    .    .    .    HUP100 EQU   $
 7866    1DA3    3A   6B   FF          LDA   MLKROW      ;SET CURSOR TO 1ST UNLK RW
 7867    1DA6    .    .    .    HUP110 EQU   $
 7868    1DA6    32   C0   FF          STA   CURROW      ;SET NEW CURRENT ROW
 7869    1DA9    AF   .    .           XRA   A
 7870    1DAA    32   C7   FF          STA   LSTROW      ;SET LAST ROW DONE TO ZERO
 7871    1DAD    57   .    .           MOV   D,A         ;SET D=0 TO FLAG TLINO UPDAT
 7872    1DAE    21   9F   FF          LXI   H,FLINE
 7873    1DB1    7E   .    .           MOV   A,M         ;SET TOP LINE POINTER TO
 7874    1DB2    CD   6E   0C          CALL  ROLUP1        ;FIRST DISPLAY LINE
 7875    1DB5    CD   76   19          CALL  CHKFMS      ;FORMAT/SOFT KEY DEFINE MODE
 7876    1DB8    C8   .    .           RZ                ;NO - RETURN
 7877    1DB9    .    .    .    ;                        YES - FALL INTO "FLDSR1" TO
 7878    1DB9    .    .    .    ;                          FIND FIRST UNPROTECTED
 7879    1DB9    .    .    .    ;                          FIELD
```

```
1DB9   .   .   .    ;*******************************************
1DB9   .   .   .    ; FLDSR - LOCATE THE NEXT UNPROTECTED FIELD *
1DB9   .   .   .    ;*******************************************
1DB9   .   .   .    ;
1DB9   .   .   .    ;  ENTRY:  DON'T CARE
1DB9   .   .   .    ;
1DB9   .   .   .    ;  EXIT :  NZ - FIELD FOUND
1DB9   .   .   .    ;            D,E = ADDRESS OF "ENDPR"
1DB9   .   .   .    ;            CURADR,CURCOL,CURROW,LSTLIN,LSTCOL
1DB9   .   .   .    ;              LSTROW UPDATE TO CORRESPOND TO
1DB9   .   .   .    ;              FIELD FOUND
1DB9   .   .   .    ;          Z - FIELD NOT FOUND
1DB9   .   .   .    ;            ALL REGISTERS DESTROYED
1DB9   .   .   .    ;
1DB9   .   .   .    FLDSR1 EQU  $          ;LOOK FOR NEXT UNPROTECT
1DB9   21  DA  FF          LXI  H,NEWROW ;INITIALIZE ROW COUNT
1DBC   36  00  .           MVI  M,0        ;TO ZERO
1DBE   2E  C1  .           MVI  L,CURCOL-BASE  ;GET CURRENT COLUMN
1DC0   4E  .   .           MOV  C,M              ;POSITION
1DC1   C3  E2  1D          JMP FSR100
1DC4   .   .   .    FLDSR  EQU $
1DC4   AF  .   .           XRA  A          ;ZERO NUMBER OF ROWS ROLLED
1DC5   32  DA  FF          STA  NEWROW
1DC8   CD  AC  06          CALL RCADRB     ;DOES CURSOR ROW EXIST?
1DCB   FA  06  07          JM   ZRETRN     ;NO - RETURN ZERO
1DCE   4F  .   .           MOV  C,A        ;YES - SAVE LAST COLUMN FOUN
1DCF   CD  65  10          CALL CKPROT     ;CURSOR IN PROTECTED FIELD?
1DD2   CA  E2  1D          JZ   FSR100     ;YES - LOOK FOR NEXT UNPROTC
1DD5   .   .   .    ;******************************************
1DD5   .   .   .    ; CURSOR IS IN UNPROTECTED FIELD           *
1DD5   .   .   .    ; SEARCH FOR START OF NEXT PROTECTED FIELD *
1DD5   .   .   .    ;******************************************
1DD5   .   .   .    FSR080 EQU $
1DD5   21  C0  C0          LXI  H,STPR*256+STPR
1DD8   CD  C4  1E          CALL FNDCU1     ;ANY MORE FIELDS IN LINE?
1DDB   CA  E8  1D          JZ   FSR120     ;NO - GO TO NEXT LINE
1DDE   .   .   .    ;*******************************************
1DDE   .   .   .    ; ADVANCE CURSOR TO START OF PROTECTED FIELD *
1DDE   .   .   .    ;*******************************************
1DDE   3E  50  .           MVI A,MAXCOL+1 ;COMPUTE NEW COLUMN
1DE0   91  .   .           SUB C
1DE1   4F  .   .           MOV C,A      ;SAVE COLUMN IN C
1DE2   .   .   .    ;*************************************
1DE2   .   .   .    ; CURSOR IS IN PROTECTED FIELD        *
1DE2   .   .   .    ; SEARCH FOR NEXT UNPROTECTED FIELD *
1DE2   .   .   .    ; IN THIS LINE                        *
1DE2   .   .   .    ;*************************************
1DE2   .   .   .    FSR100 EQU $
1DE2   CD  B9  1E          CALL FNDCHU     ;ANY MORE FIELDS IN LINE?
1DE5   C2  26  1E          JNZ  FSR200     ;YES - SET CURSOR AND DISPLA
```

```
 7932    1DE8   .   .   .   ;************************
 7933    1DE8   .   .   .   ; NO MORE FIELDS IN LINE *
 7934    1DE8   .   .   .   ; MOVE TO NEXT LINE      *
 7935    1DE8   .   .   .   ;************************
 7936    1DE8   .   .   .   FSR120 EQU $
 7937    1DE8   FE  C4  .          CPI  STPFLG     ;NON-DISPLAYING TERMINATOR?
 7938    1DEA   CA  20  1E         JZ   FSR140     ;YES - RETURN FAIL
 7939    1DED   4C  .   .          MOV  C,H        ;NO - SAVE TERMINATOR CHAR
 7940    1DEE   CD  68  19         CALL CHAINO     ;GET NEXT BLOCK LINK
 7941    1DF1   7E  .   .          MOV  A,M        ;GET NEXT LINE LINK'S MSB
 7942    1DF2   2B  .   .          DCX  H
 7943    1DF3   6E  .   .          MOV  L,M        ;PUT LSB INTO L-REGISTER
 7944    1DF4   FE  CE  .          CPI  EOP        ;END OF DISPLAY FOUND?
 7945    1DF6   CA  20  1E         JZ   FSR140     ;YES - EXIT FIELD NOT FOUND
 7946    1DF9   67  .   .          MOV  H,A        ;NO - SAVE ADDRESS OF NEW
 7947    1DFA   22  96  FF         SHLD LNKSAV       ;LINE
 7948    1DFD   EB  .   .          XCHG
 7949    1DFE   21  DA  FF         LXI  H,NEWROW  ;INCREMENT ROW NUMBER
 7950    1E01   34  .   .          INR  M
 7951    1E02   AF  .   .          XRA A
 7952    1E03   32  C6  FF         STA  LSTDCD  ;CLEAR LAST DISPLAY CODE
 7953    1E06   32  9D  FF         STA  TEMP
 7954    1E09   79  .   .          MOV A,C        ;GET LAST TERMINATOR CHAR
 7955    1E0A   0E  00  .          MVI C,0        ;SET COLUMN TO ZERO
 7956    1E0C   FE  C0  .          CPI  STPR       ;LOOKING FOR START PROTECT?
 7957    1E0E   C2  E2  1D         JNZ  FSR100     ;NO - CONTINUE UNPROTECT FIN
 7958    1E11   .   .   .   ;                      YES - SEE IF CONTINUE UNPROT
 7959    1E11   .   .   .   ;****************************************
 7960    1E11   .   .   .   ; SEARCH FOR PROTECTED FIELD            *
 7961    1E11   .   .   .   ; CHECK FOR CONTINUED UNPROTECTED FIELD *
 7962    1E11   .   .   .   ;****************************************
 7963    1E11   CD  84  1E         CALL FLDSR2     ;FIRST CHAR AN "ENDPR"
 7964    1E14   3A  9D  FF         LDA  TEMP         ;(SET NEW LSTDCD VALUE)
 7965    1E17   32  C6  FF         STA  LSTDCD
 7966    1E1A   CA  D5  1D         JZ   FSR080     ;YES - LOOK FOR START PROTEC
 7967    1E1D   C3  E2  1D         JMP  FSR100     ;NO - LOOK FOR NEXT UNPROTEC
 7968    1E20   .   .   .   ;*******************************
 7969    1E20   .   .   .   ; SET LSTCOL PAST END OF LINE   *
 7970    1E20   .   .   .   ; TO CAUSE LINE TO BE RESCANNED *
 7971    1E20   .   .   .   ;*******************************
 7972    1E20   .   .   .   FLDSRX EQU  $
 7973    1E20   .   .   .   FSR140 EQU  $              ;(Z TRUE)
 7974    1E20   21  C8  FF         LXI  H,LSTCOL
 7975    1E23   36  50  .          MVI  M,MAXCOL+1
 7976    1E25   C9  .   .          RET             ;RETURN
 7978    1E26    .    .    .    ;**************************
 7979    1E26    .    .    .    ; UNPROTECTED FIELD FOUND *
 7980    1E26    .    .    .    ; SET NEW CURSOR POSITION *
 7981    1E26    .    .    .    ;**************************
 7982    1E26    .    .    .    FSR200 EQU $
 7983    1E26    3E   50   .           MVI A,MAXCOL+1 ;COMPUTE NEW COLUMN
 7984    1E28    91   .    .           SUB C
 7985    1E29    CD   C8   21          CALL CRRET1     ;SET CURRENT CURSOR LOCATION
 7986    1E2C    32   C8   FF          STA  LSTCOL       ;AND LAST CURSOR VALUE
 7987    1E2F    EB   .    .           XCHG            ;STORE NEW CURRENT ADDRESS
 7988    1E30    22   C3   FF          SHLD CURADR
 7989    1E33    22   D5   FF          SHLD LADDR      ;SAVE FIELD ADDRESS IN
 7990    1E36    .    .    .    ;                        CASE ROLL UP NEEDED
 7991    1E36    EB   .    .           XCHG            ;RESTORE D,E AND H,L
 7992    1E37    .    .    .    ;*************************
 7993    1E37    .    .    .    ; COMPUTE NEW CURSOR ROW *
 7994    1E37    .    .    .    ;*************************
 7995    1E37    3A   DA   FF          LDA  NEWROW     ;GET NEW ABSOLUTE ROW NUMBER
 7996    1E3A    B7   .    .           ORA A      ;HAS ROW CHANGED?
 7997    1E3B    CA   7B   1E          JZ   FSR360     ;NO - RETURN
 7998    1E3E    21   C0   FF          LXI  H,CURROW   ;YES - CALCULATE NEW
 7999    1E41    86   .    .           ADD  M              ;ROW NUMBER
 8000    1E42    .    .    .    FSR240 EQU $
 8001    1E42    0E   18   .           MVI C,MAXROW+1 ;IS NEW ROW ON CURRENT PAGE?
 8002    1E44    B9   .    .           CMP C
 8003    1E45    DA   64   1E          JC   FSR340   ;YES
 8005   1E48   .   .   .   ;*************************************
 8006   1E48   .   .   .   ; NEW CURSOR ROW IS ON NEW PAGE       *
 8007   1E48   .   .   .   ; ROLL DISPLAY UP TO GET ROW ON SCREEN *
 8008   1E48   .   .   .   ;*************************************
 8009   1E48   91  .   .         SUB  C          ;DECREMENT ROLL COUNT BY ONE
 8010   1E49   21  6B  FF        LXI  H,MLKROW  ;PAGE
 8011   1E4C   86  .   .         ADD  M          ;ADJUST FOR LOCKED DISPLAY
 8012   1E4D   57  .   .         MOV  D,A        ;SAVE RESULT FOR STORAGE
 8013   1E4E   79  .   .         MOV  A,C        ;COMPUTE NUMBER OF LINES TO
 8014   1E4F   96  .   .         SUB  M            ;ROLL FOR ONE PAGE
 8015   1E50   5F  .   .         MOV  E,A        ;SAVE THE VALUE FOR STORAGE
 8016   1E51   EB  .   .         XCHG            ;PUT VALUES INTO H,L
 8017   1E52   22  82  FF        SHLD ROLLCT  ;STORE ROLL PARAMETERS
 8018   1E55   .   .   .   ;
 8019   1E55   .   .   .   ;  ROLL UP ONE PAGE OF LINES
 8020   1E55   .   .   .   ;
 8021   1E55   .   .   .   FSR300 EQU  $
 8022   1E55   CD  27  0C        CALL ROLLUP     ;ROLLUP ONE LINE
 8023   1E58   21  82  FF        LXI  H,ROLLCT
 8024   1E5B   35  .   .         DCR  M          ;PAGE ROLLED UP?
 8025   1E5C   C2  55  1E        JNZ  FSR300     ;NO - DO ANOTHER LINE
 8026   1E5F   23  .   .         INX  H          ;YES - GET NUMBER OF ROWS
 8027   1E60   7E  .   .         MOV  A,M          ;TO UNPROTECTED FIELD AND
 8028   1E61   C3  42  1E        JMP  FSR240       ;CHECK TO SEE IF ON SCREEN
 8029   1E64   .   .   .   ;*************
 8030   1E64   .   .   .   ; UPDATE ROW *
 8031   1E64   .   .   .   ;*************
 8032   1E64   .   .   .   FSR340 EQU  $
 8033   1E64   32  C0  FF        STA  CURROW     ;SET NEW ROW NUMBER
 8034   1E67   2A  C0  FF        LHLD CURROW     ;SET LAST ROW AND COLUMN DON
 8035   1E6A   22  C7  FF        SHLD LSTROW       ;CURRENT ROW AND COLUMN
 8036   1E6D   2A  96  FF        LHLD LNKSAV     ;SET "LSTLIN" TO CURRENT ROW
 8037   1E70   23  .   .         INX  H            ;ADDRESS
 8038   1E71   22  C9  FF        SHLD LSTLIN
 8039   1E74   2A  D5  FF        LHLD LADDR      ;SET "CURADR" TO ADDRESS OF
 8040   1E77   22  C3  FF        SHLD CURADR       ;FIRST CHAR IN NEW FIELD
 8041   1E7A   EB  .   .         XCHG            ;PUT CURRENT ADDRESS INTO D,
 8042   1E7B   .   .   .   FSR360 EQU  $
 8043   1E7B   FE  44  .         CPI  D          ;SET Z-FALSE (D >= 320)
 8044   1E7D   C3  9E  0F        JMP  DISLN1     ;GO SET DISPLAY CURSOR ROW
```

```
=========================================================================
 ITEM     LOC    OBJECT CODE   SOURCE STATEMENTS
=========================================================================
 8046    1E80    .    .    .   ;
 8047    1E80    .    .    .   ; * * * * * * * * * * * * * * * * * * * * * * * *
 8048    1E80    .    .    .   ;
 8049    1E80    .    .    .   ;  FLDSR2 - DETERMINE PROTECT SENSE OF NEXT
 8050    1E80    .    .    .   ;    CHARACTER
 8051    1E80    .    .    .   ;
 8052    1E80    .    .    .   ;      ENTRY:  D,E = NEXT CHARACTER ADDRESS
 8053    1E80    .    .    .   ;
 8054    1E80    .    .    .   ;      EXIT :  Z - CONTINUATION OF FORMAT FIELD
 8055    1E80    .    .    .   ;              NZ - NOT A CONTINUATION
 8056    1E80    .    .    .   ;              D,E = ADDRESS OF CHARACTER
 8057    1E80    .    .    .   ;              H = BASEH
 8058    1E80    .    .    .   ;              TEMP = NEW ENHANCEMENT CODE IF ANY
 8059    1E80    .    .    .   ;              A,L DESTROYED
 8060    1E80    .    .    .   ;
 8061    1E80    .    .    .   FS2000 EQU  $
 8062    1E80    32   9D   FF         STA  TEMP  ;STORE NEW DISPLAY CONTROL
 8063    1E83    .    .    .   FLDSRB EQU  $
 8064    1E83    .    .    .   FS2005 EQU  $
 8065    1E83    1B   .    .          DCX  D          ;SET ADDRESS TO NEXT CHAR
 8066    1E84    .    .    .   FLDSR2 EQU $
 8067    1E84    13   .    .          INX  D          ;SET ADDRESS TO PREV CHAR
 8068    1E85    CD   87   0B         CALL NXTCHR  ;GET NEXT CHARACTER
 8069    1E88    C2   84   1E         JNZ FLDSR2   ;SKIP OVER LINKS
 8070    1E8B    87   .    .          ADD  A          ;ASCII OR DISPLAY CONTROL?
 8071    1E8C    D2   01   0B         JNC  NZEXIT     ;ASCII - RETURN NOT CONTINUE
 8072    1E8F    1F   .    .          RAR               ;(RESTORE DATA BYTE)
 8073    1E90    F2   80   1E         JP   FS2000     ;DISPLAY CONTROL - IGNORE IT
 8074    1E93    FE   C4   .          CPI  STPFLG     ;TERMINATOR OR TYPE DEFINE?
 8075    1E95    F2   83   1E         JP   FS2005     ;YES - SKIP TO NEXT CHARACTE
 8076    1E98    21   C5   FF         LXI  H,LSTFMT   ;COMPARE AGAINST LAST FORMAT
 8077    1E9B    BE   .    .          CMP  M            ;CONTROL AND RETURN
 8078    1E9C    C9   .    .          RET
```

```
8080    1E9D    .    .    .    ;
8081    1E9D    .    .    .    ; * * * * * * * * * * * * * * * * * * * * * * *
8082    1E9D    .    .    .    ;
8083    1E9D    .    .    .    ;  FNDCH - SEE IF NEXT CHAR IS FORMAT CONTROL BYTE
8084    1E9D    .    .    .    ;
8085    1E9D    .    .    .    ;      ENTRY:  TERMINAL IS IN FORMAT MODE
8086    1E9D    .    .    .    ;              D,E = START ADDRESS
8087    1E9D    .    .    .    ;              H,L = CHARACTERS TO LOOK FOR
8088    1E9D    .    .    .    ;
8089    1E9D    .    .    .    ;      EXIT :  Z - CHARACTER NOT FOUND
8090    1E9D    .    .    .    ;              NZ - CHARACTER FOUND
8091    1E9D    .    .    .    ;              D,E = ADDRESS OF ENDING CHARACTER
8092    1E9D    .    .    .    ;              A,B,C,L,TEMP DESTROYED
8093    1E9D    .    .    .    ;
8094    1E9D    .    .    .    ;  FNDCHO - SEE IF NEXT CHARACTER IS PROTECTED
8095    1E9D    .    .    .    ;
8096    1E9D    .    .    .    FNDCHO EQU   $
8097    1E9D    21   C0   C0          LXI  H,STPR*256+STPR  ;SET COMPARE CHARS
8098    1EA0    .    .    .    FNDCH  EQU   $
8099    1EA0    3E   01   .           MVI  A,IGNTRM  ;SET TO IGNORE NON-DISPLAYIN
8100    1EA2    32   6D   FF          STA  TRMFCT      ;TERMINATOR
8101    1EA5    3A   C2   FF          LDA  PROFLD    ;SAVE PROTECTED FIELD
8102    1EA8    F5   .    .           PUSH PSW         ;STATUS
8103    1EA9    0E   00   .           MVI  C,0       ;SET FOR NEXT CHARACTER ONLY
8104    1EAB    CD   C8   1E          CALL FCR400    ;LOCATE THE NEXT CHARACTER
8105    1EAE    3E   00   .           MVI  A,DELTRM  ;RESTORE FLAG TO DELETE NON-
8106    1EB0    32   6D   FF          STA  TRMFCT      ;DISPLAYING TERMINATOR
8107    1EB3    C1   .    .           POP  B         ;RESET PROTECT STATUS TO BE
8108    1EB4    78   .    .           MOV  A,B         ;CONSISTENT WITH CHARACTER
8109    1EB5    32   C2   FF          STA  PROFLD      ;POINTED TO BY "CURADR"
8110    1EB8    C9   .    .           RET            ;RETURN
```

```
1EB9    .    .    .     ;
1EB9    .    .    .     ; * * * * * * * * * * * * * * * * * * * * * * * *
1EB9    .    .    .     ;
1EB9    .    .    .     ;  FNDCHU - LOCATE NEXT UNPROTECTED FIELD
1EB9    .    .    .     ;    CONTROL BYTE IN CURRENT LINE
1EB9    .    .    .     ;
1EB9    .    .    .     ;      ENTRY:  TERMINAL IS IN FORMAT MODE
1EB9    .    .    .     ;              B = DON'T CARE
1EB9    .    .    .     ;              C = CURRENT COLUMN NUMBER
1EB9    .    .    .     ;              D,E = START ADDRESS
1EB9    .    .    .     ;
1EB9    .    .    .     ;      EXIT :  Z - CHARACTER NOT FOUND
1EB9    .    .    .     ;              NZ - CHARACTER FOUND
1EB9    .    .    .     ;              C = NUMBER OF CHARS TO END OF LINE
1EB9    .    .    .     ;              D,E = ADDRESS OF ENDING CHARACTER
1EB9    .    .    .     ;              PROFLD SET AS DEFINED
1EB9    .    .    .     ;              A,B,L DESTROYED
1EB9    .    .    .     ;
1EB9    .    .    .     FNDCHU EQU  $
1EB9    CD   A6   12          CALL DCXB2D    ;DATA COMM OR I/O BUFF INPUT
1EBC    21   C1   C1          LXI  H,ENDPR*256+ENDPR  ;(SET "ENDPR" ONLY)
1EBF    CA   C4   1E          JZ   FNDCU1    ;NO - SKIP XMIT ONLY FIELDS
1EC2    2E   C2   .           MVI  L,XMONLY  ;YES - LOOK FOR "XMONLY" ALS
1EC4    .    .    .     ;
1EC4    .    .    .     ;  LOCATE THE FORMAT CONTROL CHARACTER
1EC4    .    .    .     ;
1EC4    .    .    .     FNDCU1 EQU  $
1EC4    3E   4F   .           MVI A,MAXCOL ;COMPUTE NO. OF CHARS
1EC6    91   .    .           SUB C        ;TO SEARCH
1EC7    4F   .    .           MOV C,A
1EC8    .    .    .     FCR400 EQU $
1EC8    CD   CF   1E          CALL FNDCHR      ;LOOK FOR SPECIFIED CHARS
1ECB    C8   .    .           RZ               ;RETURN IF EOL ENCOUNTERED
1ECC    AF   .    .           XRA  A           ;OTHERWISE, SET FLAG TO
1ECD    B1   .    .           ORA  C             ;SHOW IF CHARACTER FOUND
1ECE    C9   .    .           RET
```

```
==============================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                          PAGE 243
==============================================================================
8149   1ECF    .    .    .   ;***************************************
8150   1ECF    .    .    .   ; FNDCHR - LOCATE SPECIFIED CHARACTER *
8151   1ECF    .    .    .   ;***************************************
8152   1ECF    .    .    .   ;
8153   1ECF    .    .    .   ;  ENTRY:  C = NUMBER OF COLUMNS TO SEARCH
8154   1ECF    .    .    .   ;          D,E = STARTING ADDRESS
8155   1ECF    .    .    .   ;          H,L = CHARACTERS TO LOOK FOR
8156   1ECF    .    .    .   ;            (VALID FOR FORMAT MODE ONLY)
8157   1ECF    .    .    .   ;
8158   1ECF    .    .    .   ;  EXIT :  Z - CHARACTER NOT FOUND
8159   1ECF    .    .    .   ;          NZ - CHARACTER FOUND
8160   1ECF    .    .    .   ;          C = NUMBER OF CHARACTERS LEFT
8161   1ECF    .    .    .   ;            (= 0, IF CHARACTER FOUND)
8162   1ECF    .    .    .   ;          D,E = ADDRESS OF TERMINATING CHARACTER
8163   1ECF    .    .    .   ;          "EOLMV" SET TO ZERO
8164   1ECF    .    .    .   ;          "PROFLD" SET IF IN FORMAT MODE
8165   1ECF    .    .    .   ;          "LSTFMT" UPDATED IF A FORMAT CONTROL
8166   1ECF    .    .    .   ;            CHARACTER IS ENCOUNTERED
8167   1ECF    .    .    .   ;
8168   1ECF    .    .    .   FNDCHR EQU  $
8169   1ECF   AF    .    .          XRA  A
8170   1ED0   32   90   FF          STA  EOLMV
8171   1ED3   13    .    .          INX  D           ;SET TO PREV CHAR ADDRESS
8172   1ED4   0C    .    .          INR  C           ;ADJUST CHARACTER COUNT
8173   1ED5   0C    .    .          INR  C
8174   1ED6    .    .    .   FCR005 EQU  $
8175   1ED6   0D    .    .          DCR  C           ;COLUMN FOUND?
8176   1ED7   CA   01   0B          JZ   NZEXIT      ;YES - RETURN CHARACTER FOUN
8177   1EDA    .    .    .   ;
8178   1EDA    .    .    .   ;  SEARCH DISPLAY LIST
8179   1EDA    .    .    .   ;
8180   1EDA    .    .    .   FCR010 EQU  $
8181   1EDA   CD   87   0B          CALL NXTCHR      ;GET THE NEXT CHARACTER
8182   1EDD   C2   56   1F          JNZ  FCR260      ;EOL LINK - EXIT NOT FOUND
8183   1EE0   87    .    .          ADD  A           ;IS IT ASCII?
8184   1EE1   D2   D6   1E          JNC  FCR005      ;YES - DECREMENT COLUMN COUN
8185   1EE4    .    .    .   ;**********************************************
8186   1EE4    .    .    .   ; NON-ASCII CHARACTER - DETERMINE CHAR FUNCTION *
8187   1EE4    .    .    .   ;**********************************************
8188   1EE4   1F    .    .          RAR              ;RESTORE CHARACTER
8189   1EE5   FA   EE   1E          JM   FCR100      ;NOT DISPLAY CTL - CHECK MOR
8190   1EE8   32   C6   FF          STA  LSTDCD      ;UPDATE CURRENT DISPLAY CODE
8191   1EEB   C3   DA   1E          JMP  FCR010      ;CONTINUE SEARCHING
```

```
 8193   1EEE   .   .   .   ;
 8194   1EEE   .   .   .   ;  FORMAT CONTROL CHARACTER - CHECK FOR ENDING
 8195   1EEE   .   .   .   ;
 8196   1EEE   .   .   .   FCR100 EQU  $
 8197   1EEE   FE  CC  .          CPI  EOL          ;END OF LINE?
 8198   1EF0   C8  .   .          RZ                ;YES - RETURN
 8199   1EF1   FE  CE  .          CPI  EOP          ;END OF DISPLAY?
 8200   1EF3   C8  .   .          RZ                ;YES - RETURN
 8201   1EF4   FE  C4  .          CPI  STPFLG       ;NON-DISPLAYING TERMINATOR?
 8202   1EF6   CA  45  1F         JZ   FCR200       ;YES - DETERMINE ITS FUNCTIO
 8203   1EF9   FE  C5  .          CPI  ALPHA        ;TYPE DEFINITION?
 8204   1EFB   F2  26  1F         JP   FCR150       ;YES - SET CHECK FUNCTION
 8205   1EFE   FE  C3  .          CPI  XMONLY+1     ;FORMAT CONTROL?
 8206   1F00   F2  DA  1E         JP   FCR010       ;NO - CONTINUE SEARCHING
 8207   1F03   E5  .   .          PUSH H            ;YES - RESET CHECK ROUTINE
 8208   1F04   21  06  07         LXI  H,ZRETRN       ;ADDRESS
 8209   1F07   22  86  FF         SHLD CHKRTN
 8210   1F0A   E1  .   .          POP  H            ;RESTORE CHECK CHARACTERS
 8211   1F0B   32  C5  FF         STA  LSTFMT       ;SET CURRENT FORMAT CONTROL
 8212   1F0E   47  .   .          MOV  B,A          ;SAVE CONTROL CHARACTER
 8213   1F0F   CD  76  19         CALL CHKFMS       ;FORMAT/SOFT KEY DEFINE MODE
 8214   1F12   CA  DA  1E         JZ   FCR010       ;NO - CONTINUE SEARCHING
 8215   1F15   78  .   .          MOV  A,B          ;RECALL CHARACTER
 8216   1F16   DE  C1  .          SBI  STPR+1       ;COMPUTE "PROFLD" VALUE
 8217   1F18   32  C2  FF         STA  PROFLD         ;(= -1 FOR PROTECTED)
 8218   1F1B   78  .   .          MOV  A,B          ;RECALL CHARACTER
 8219   1F1C   BC  .   .          CMP  H            ;TERMINATOR FOUND?
 8220   1F1D   CA  24  1F         JZ   FCR110       ;YES - EXIT
 8221   1F20   BD  .   .          CMP  L
 8222   1F21   C2  DA  1E         JNZ  FCR010       ;NO - CONTINUE SEARCHING
 8223   1F24   .   .   .   FCR110 EQU  $
 8224   1F24   B7  .   .          ORA  A            ;SET Z FALSE
 8225   1F25   C9  .   .          RET               ;RETURN
 8226   1F26   .   .   .   ;
 8227   1F26   .   .   .   ;  TYPE DEFINITION FOUND - SET CHECK ROUTINE
 8228   1F26   .   .   .   ;
 8229   1F26   .   .   .   FCR150 EQU  $
 8230   1F26   E5  .   .          PUSH H            ;SAVE TERMINATOR CHARACTERS
 8231   1F27   21  23  48         LXI  H,ZALPCK     ;SET H,L FOR ALPHA CHECK
 8232   1F2A   CA  3E  1F         JZ   FCR160       ;SET ALPHA CHECK IF ALPHA
 8233   1F2D   21  26  48         LXI  H,ZNUMCK     ;SET H,L FOR NUMERIC CHECK
 8234   1F30   D6  C7  .          SUI  NUMBER+1     ;NUMERIC FIELD?
 8235   1F32   FA  3E  1F         JM   FCR160       ;YES - SET CHECK ROUTINE ADD
 8236   1F35   21  06  07         LXI  H,ZRETRN     ;NO - SET H,L FOR ALPHANUM
 8237   1F38   CA  3E  1F         JZ   FCR160       ;SET ROUTINE ADDR IF = ZERO
 8238   1F3B   21  B7  0F         LXI  H,SFKCHK       ;ELSE, SET FOR SOFT KEYS
 8239   1F3E   .   .   .   FCR160 EQU  $
 8240   1F3E   22  86  FF         SHLD CHKRTN       ;SET CHECK ROUTINE ADDRESS
 8241   1F41   E1  .   .          POP  H            ;RECALL TERMINATOR CHARACTER
 8242   1F42   C3  DA  1E         JMP  FCR010       ;CONTINUE SEARCHING
 8244   1F45   .   .   .   ;*********************************************
 8245   1F45   .   .   .   ; NON-DISPLAYING TERMINATOR FOUND - DETERMINE *
 8246   1F45   .   .   .   ;   AND PERFORM ITS FUNCTION                  *
 8247   1F45   .   .   .   ;*********************************************
 8248   1F45   .   .   .   FCR200 EQU  $
 8249   1F45   3A  6D  FF         LDA  TRMFCT      ;GET THE FUNCTION FLAG
 8250   1F48   B7  .   .          ORA  A           ;WHAT FUNCTION?
 8251   1F49   FA  55  1F         JM   FCR250      ;-1 - TERMINATE TRANSFER
 8252   1F4C   C2  DA  1E         JNZ  FCR010      ;+1 - IGNORE IT
 8253   1F4F   CD  B8  1A         CALL CHRDL2       ;0 - DELETE IT
 8254   1F52   C3  DA  1E         JMP  FCR010      ;CONTINUE CHARACTER SEARCH
 8255   1F55   .   .   .   ;
 8256   1F55   .   .   .   ;  TERMINATE TRANSFER
 8257   1F55   .   .   .   ;
 8258   1F55   .   .   .   FCR250 EQU  $
 8259   1F55   1A  .   .          LDAX D           ;PUT CHARACTER BACK IN A-REG
 8260   1F56   .   .   .   FCR260 EQU  $
 8261   1F56   BF  .   .          CMP  A           ;SET Z-FLAG TRUE
 8262   1F57   C9  .   .          RET              ;RETURN CHARACTER NOT FOUND
```

```
8264   1F58   .   .   .   ;*********************************************
8265   1F58   .   .   .   ; FNDLST - LOCATE LAST CHARACTER TYPE AHEAD OF *
8266   1F58   .   .   .   ;   CURRENT CHARACTER                          *
8267   1F58   .   .   .   ;*********************************************
8268   1F58   .   .   .   ;
8269   1F58   .   .   .   ;  ENTRY:  A = NUMBER OF COLUMNS TO SEARCH
8270   1F58   .   .   .   ;          D,E = ADDRESS OF CHARACTER BEFORE
8271   1F58   .   .   .   ;            BEFORE FIRST CHARACTER TO LOOK AT
8272   1F58   .   .   .   ;          H,L = CHARACTERS TO BE FOUND
8273   1F58   .   .   .   ;
8274   1F58   .   .   .   ;  EXIT :  P - CHARACTER FOUND
8275   1F58   .   .   .   ;            B = NUMBER OF CHARACTERS FROM CURRENT
8276   1F58   .   .   .   ;              CHARACTER
8277   1F58   .   .   .   ;          M - CHARACTER NOT FOUND
8278   1F58   .   .   .   ;            B DESTROYED
8279   1F58   .   .   .   ;          A,C,D,E DESTROYED
8280   1F58   .   .   .   ;
8281   1F58   .   .   .   FNDLS0 EQU  $
8282   1F58   3C  .   .          INR  A             ;ADJUST SEARCH COUNT
8283   1F59   21  CC  CC         LXI  H,EOL*256+EOL  ;SET TO LOOK FOR "EOL"
8284   1F5C   .   .   .   ;
8285   1F5C   .   .   .   FNDLST EQU  $
8286   1F5C   4F  .   .          MOV  C,A           ;PUT SEARCH COUNT IN C-REG
8287   1F5D   06  FF  .          MVI  B,377Q        ;PRESET B FOR FAIL RETURN
8288   1F5F   3D  .   .          DCR  A             ;ANY COLUMNS TO SEARCH?
8289   1F60   F8  .   .          RM                 ;NO - RETURN NONE FOUND
8290   1F61   .   .   .   FLS010 EQU  $
8291   1F61   CD  87  08         CALL NXTCHR        ;GET THE NEXT CHARACTER
8292   1F64   BC  .   .          CMP  H             ;DOES IT MATCH DESIRED CHARS
8293   1F65   CA  6C  1F         JZ   FLS020        ;YES - SAVE LOCATION OF CHAR
8294   1F68   BD  .   .          CMP  L
8295   1F69   C2  6D  1F         JNZ  FLS030        ;NO - GO TO NEXT CHARACTER
8296   1F6C   .   .   .   FLS020 EQU  $
8297   1F6C   41  .   .          MOV  B,C           ;SAVE LOCATION OF CHAR IN B
8298   1F6D   .   .   .   FLS030 EQU  $
8299   1F6D   B7  .   .          ORA  A             ;IS CURRENT CHAR ASCII?
8300   1F6E   FA  78  1F         JM   FLS050        ;NO - CHECK FOR TERMINATION
8301   1F71   0D  .   .          DCR  C             ;SEARCH COMPLETE?
8302   1F72   .   .   .   FLS035 EQU  $
8303   1F72   C2  61  1F         JNZ  FLS010        ;NO - CHECK NEXT CHARACTER
8304   1F75   .   .   .   FLS040 EQU  $
8305   1F75   AF  .   .          XRA  A             ;CLEAR A-REGISTER
8306   1F76   B0  .   .          ORA  B             ;SET FLAGS FOR RETURN
8307   1F77   C9  .   .          RET                ;RETURN
8308   1F78   .   .   .   ;*********************************************
8309   1F78   .   .   .   ; NON-ASCII CHARACTER - CHECK FOR TERMINATION *
8310   1F78   .   .   .   ;*********************************************
8311   1F78   .   .   .   FLS050 EQU  $
8312   1F78   FE  CC  .          CPI  EOL           ;IS IT AN EOL?
8313   1F7A   CA  75  1F         JZ   FLS040        ;YES - EXIT
8314   1F7D   FE  CE  .          CPI  EOP           ;IS IT AN EOP?
8315   1F7F   C3  72  1F         JMP  FLS035        ;GO CHECK RESULT
```

```
================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                          PAGE 247
================================================================================
 8317   1F82   .    .    .   ;***********************************
 8318   1F82   .    .    .   ; HTAB - SKIP TO NEXT TAB POSITION *
 8319   1F82   .    .    .   ;***********************************
 8320   1F82   .    .    .   HTAB   EQU  $
 8321   1F82   CD   72   19         CALL CHKFMO    ;FORMAT/SOFT KEY DEFINE MODE
 8322   1F85   C2   CE   1F         JNZ  HTB200    ;YES - LOCATE NEXT FIELD
 8323   1F88   2E   C1   .          MVI  L,CURCOL-BASE  ;NO - LOCATE NEXT TAB
 8324   1F8A   46   .    .          MOV  B,M           ;SET POSITION
 8325   1F8B   04   .    .          INR  B          ;START FROM NEXT COLUMN
 8326   1F8C   3E   4F   .          MVI  A,MAXCOL  ;COMPUTE NUMBER OF COLUMNS
 8327   1F8E   90   .    .          SUB  B            ;TO END OF LINE
 8328   1F8F   FA   96   20         JM   CRLF       ;GO TO START OF NEXT LINE IF
 8329   1F92   .    .    .   ;                        ALREADY AT END OF LINE
 8330   1F92   F6   07   .          ORI 7         ;MOVE TO COL CORRESP. TO
 8331   1F94   .    .    .   ;                     START OF BYTE
 8332   1F94   4F   .    .          MOV C,A       ;SAVE IN C
 8333   1F95   78   .    .          MOV A,B
 8334   1F96   CD   ED   14         CALL FNDTB1  ;GET TABLE ENTRY FOR COLUMN
 8335   1F99   3D   .    .          DCR  A          ;MASK OFF BITS FOR
 8336   1F9A   2F   .    .          CMA                ;PREVIOUS COLUMNS
 8337   1F9B   A6   .    .          ANA M
 8338   1F9C   .    .    .   ;*********************************
 8339   1F9C   .    .    .   ; CHECK NEXT COLUMN FOR SET TAB *
 8340   1F9C   .    .    .   ;*********************************
 8341   1F9C   .    .    .   HTB100 EQU $
 8342   1F9C   06   08   .          MVI B,8       ;GET BIT COUNT
 8343   1F9E   CA   C1   1F         JZ  HTB140   ;NO BITS SET IN BYTE
 8344   1FA1   .    .    .   HTB120 EQU  $
 8345   1FA1   0F   .    .          RRC             ;TAB BIT SET?
 8346   1FA2   D2   BD   1F         JNC  HTB130    ;NO - TRY NEXT COLUMN
 8347   1FA5   .    .    .   ;*****************************
 8348   1FA5   .    .    .   ; TAB IS SET - UPDATE CURCOL *
 8349   1FA5   .    .    .   ;*****************************
 8350   1FA5   .    .    .   HTB160 EQU  $
 8351   1FA5   5F   .    .          MOV  E,A        ;SAVE A-REGISTER
 8352   1FA6   3E   56   .          MVI  A,MAXCOL+7  ;COMPUTE COLUMN OF LOCATIO
 8353   1FA8   91   .    .          SUB  C              ;OF TAB
 8354   1FA9   90   .    .          SUB B
 8355   1FAA   22   96   FF         SHLD LNKSAV     ;SAVE CURRENT TABLE ADDRESS
 8356   1FAD   2A   BE   FF         LHLD RHTMGN     ;GET RIGHT AND LEFT MARGINS
 8357   1FB0   BD   .    .          CMP  L          ;TAB BEYOND RIGHT MARGIN?
 8358   1FB1   F2   96   20         JP   CRLF       ;YES - DO CR, LF
 8359   1FB4   3C   .    .          INR  A          ;NO - ADJUST TO PROPER VALUE
 8360   1FB5   BC   .    .          CMP  H          ;TAB BEYOND LEFT MARGIN?
 8361   1FB6   D2   98   11         JNC  CURP04     ;YES - LOCATE TAB LOCATION
 8362   1FB9   7B   .    .          MOV  A,E        ;NO - RESTORE A-REGISTER
 8363   1FBA   2A   96   FF         LHLD LNKSAV     ;RECALL TAB TABLE ADDRESS
 8364   1FBD   .    .    .   ;                      LOOK FOR ANOTHER TAB
 8366    1FBD    .    .    .    ;***********************************
 8367    1FBD    .    .    .    ; TAB NOT FOUND - CHECK NEXT COLUMN *
 8368    1FBD    .    .    .    ;***********************************
 8369    1FBD    .    .    .    HTB130 EQU  $          ;NO - TRY NEXT COLUMN
 8370    1FBD    05   .    .           DCR  B          ;ALL BITS EXAMINED?
 8371    1FBE    C2   A1   1F          JNZ  HTB120     ;NO - LOOK TO NEXT BIT
 8372    1FC1    .    .    .    ;****************************
 8373    1FC1    .    .    .    ; BYTE EXHAUSTED            *
 8374    1FC1    .    .    .    ; MOVE TO NEXT TABIBL ENTRY *
 8375    1FC1    .    .    .    ;****************************
 8376    1FC1    .    .    .    HTB140 EQU $
 8377    1FC1    79   .    .           MOV A,C      ;GET COLUMN COUNT
 8378    1FC2    D6   08   .           SUI 8        ;DECREMENT
 8379    1FC4    FA   96   20          JM   CRLF       ;DO CR,LF IF REACHED END
 8380    1FC7    4F   .    .           MOV C,A
 8381    1FC8    23   .    .           INX  H          ;GET NEXT BYTE FROM TABLE
 8382    1FC9    7E   .    .           MOV A,M
 8383    1FCA    B7   .    .           ORA A        ;SET FLAGS
 8384    1FCB    C3   9C   1F          JMP HTB100
```

```
==============================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS
==============================================================================
8386   1FCE    .    .    .   ;******************
8387   1FCE    .    .    .   ; FORMAT MODE TAB *
8388   1FCE    .    .    .   ;******************
8389   1FCE    .    .    .   HTB200 EQU $
8390   1FCE    CD   C4   1D         CALL FLDSR    ;SEARCH FOR NEXT FIELD
8391   1FD1    C0   .    .          RNZ           ;RETURN IF FOUND
8392   1FD2    C3   2C   1D         JMP  CURPH1     ;HOME TO FIRST UNPROT. FIELD
```

```
=============================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                      PAGE 250
=============================================================================
 8394    1FD5    .    .    .    ;
 8395    1FD5    .    .    .    ; * * * * * * * * * * * * * * * * * * * * * * * *
 8396    1FD5    .    .    .    ;
 8397    1FD5    .    .    .    ;  ICHON,ICHOFF - INSERT CHARACTER ON/OFF
 8398    1FD5    .    .    .    ;
 8399    1FD5    .    .    .    ICHON  EQU  $
 8400    1FD5    06   00   .           MVI  B,0        ;SET FOR NO BLINK
 8401    1FD7    .    .    .    ICH010 EQU  $
 8402    1FD7    3E   02   .           MVI  A,INSCHR   ;TURN ON INSERT CHARACTER
 8403    1FD9    C3   0E   48          JMP  ZSTMD1       ;LED AND EXIT
 8404    1FDC    .    .    .    ;
 8405    1FDC    .    .    .    ICHOFF EQU  $
 8406    1FDC    3E   FD   .           MVI  A,377Q-INSWRP
 8407    1FDE    CD   DC   13          CALL CLCMFL     ;CLEAR WRAP AROUND FLAG
 8408    1FE1    3E   02   .           MVI  A,INSCHR   ;TURN OFF INSERT CHARACTER
 8409    1FE3    C3   11   48          JMP  ZCLMD1
 8410    1FE6    .    .    .    ;*************************************
 8411    1FE6    .    .    .    ; IWRPON - INSERT WITH WRAPAROUND ON *
 8412    1FE6    .    .    .    ;*************************************
 8413    1FE6    .    .    .    IWRPON EQU  $
 8414    1FE6    3E   02   .           MVI  A,INSWRP
 8415    1FE8    CD   00   14          CALL STCMFL     ;SET WRAP AROUND FLAG
 8416    1FEB    06   FF   .           MVI  B,377Q     ;SET TO BLINK LED
 8417    1FED    C3   D7   1F          JMP  ICH010     ;SET INSERT CHARACTER LED ON
 8419   1FF0    .    .    .   ;*********************************************
 8420   1FF0    .    .    .   ; BCKSPC - BACKSPACE ONE CHARACTER POSITION *
 8421   1FF0    .    .    .   ;*********************************************
 8422   1FF0    .    .    .   BCKSPC EQU  $
 8423   1FF0   2E   C1    .          MVI  L,CURCOL-BASE
 8424   1FF2   35    .    .          DCR  M           ;DECREMENT CURRENT COLUMN
 8425   1FF3   F0    .    .          RP               ;RETURN IF NOT AT COLUMN ZER
 8426   1FF4   34    .    .          INR  M           ;ELSE, RESTORE TO ZERO AND
 8427   1FF5   C9    .    .          RET              ;RETURN
 8429   1FF6    .   .   .   ;**************************
 8430   1FF6    .   .   .   ;  R O M   B R E A K   4  *
 8431   1FF6    .   .   .   ;**************************
 8432   1FF6    .   .   .          ORG  ZBRK3+4000Q
 8433   2000    .   .   .   ZBRK4  EQU  $
 8434   2000   50   .   .          DB   VERSN     ;ROM PRESENT FLAGS
 8435   2001   20   .   .          DB   ZBRK4/256
```

```
8437   2002   .   .   .   ;**********************************
8438   2002   .   .   .   ; CURADV - CURSOR ADVANCE ROUTINE  *
8439   2002   .   .   .   ; ADVANCES CURSOR TO NEXT POSITION *
8440   2002   .   .   .   ; ON DISPLAY                       *
8441   2002   .   .   .   ;**********************************
8442   2002   .   .   .   CURAD2 EQU  $         ;ADVANCE CURSOR TWICE
8443   2002   CD  05  20         CALL CURADV    ;DO FIRST CURSOR ADVANCE
8444   2005   .   .   .   ;                      THEN FALL IN TO DO NEXT
8445   2005   .   .   .   CURADV EQU $
8446   2005   CD  57  20         CALL CRADV   ;ADVANCE CURSOR
8447   2008   CD  76  19         CALL CHKFMS    ;FORMAT/SOFT KEY DEFINE MODE
8448   200B   C8  .   .          RZ             ;NO - RETURN
8449   200C   .   .   .   ;****************************************
8450   200C   .   .   .   ; FORMAT MODE                           *
8451   200C   .   .   .   ; CHECK FOR ADVANCE INTO PROTECTED FIELD *
8452   200C   .   .   .   ;****************************************
8453   200C   3A  C1  FF         LDA  CURCOL    ;GET NEW CURRENT COLUMN
8454   200F   B7  .   .          ORA  A         ;DID CURSOR WRAP AROUND?
8455   2010   C2  28  20         JNZ  CRA040    ;NO - CHECK FOR PROTECTED FL
8456   2013   .   .   .   ;**********************************
8457   2013   .   .   .   ; CURSOR WRAPPED AROUND            *
8458   2013   .   .   .   ; SEE IF NEW LINE IS CONTINUATION *
8459   2013   .   .   .   ; OF UNPROTECTED FIELD             *
8460   2013   .   .   .   ;**********************************
8461   2013   2A  C9  FF         LHLD LSTLIN
8462   2016   EB  .   .          XCHG           ;GET CURRENT LINE ADDR IN D,
8463   2017   3A  8A  FF         LDA  FMTCTL    ;RESET "LSTFMT" TO LAST
8464   201A   32  C5  FF         STA  LSTFMT      ;FORMAT CONTROL IN LINE
8465   201D   CD  83  1E         CALL FLDSRB    ;CONTINUATION FIELD?
8466   2020   C2  39  20         JNZ  CRA060    ;NO - TAB TO NEXT FIELD
8467   2023   .   .   .   ;*******************
8468   2023   .   .   .   ; RESET CURADV FLAG *
8469   2023   .   .   .   ;*******************
8470   2023   .   .   .   CRADV1 EQU $
8471   2023   AF  .   .          XRA  A
8472   2024   32  67  FF         STA  CRAFLG
8473   2027   C9  .   .          RET
```

```
 8475   2028    .   .   .   ;****************************
 8476   2028    .   .   .   ; CURSOR DID NOT WRAP AROUND *
 8477   2028    .   .   .   ; SEE IF CURSOR ENTERED      *
 8478   2028    .   .   .   ;  PROTECTED FIELD           *
 8479   2028    .   .   .   ;****************************
 8480   2028    .   .   .   CRA040 EQU  $
 8481   2028    2A  C3  FF         LHLD CURADR  ;GET THE CURRENT CHAR ADDR
 8482   202B    EB  .   .          XCHG            ;PUT IT INTO H,L
 8483   202C    1B  .   .          DCX  D          ;SET POINTER TO NEXT CHAR
 8484   202D    2A  86  FF         LHLD CHKRTN     ;SAVE THE CURRENT CHECK
 8485   2030    E5  .   .          PUSH H            ;ROUTINE ADDRESS
 8486   2031    CD  9D  1E         CALL FNDCH0     ;NEXT CHARACTER PROTECTED?
 8487   2034    E1  .   .          POP  H            ;(RESTURE CHECK ROUTINE
 8488   2035    22  86  FF         SHLD CHKRTN          ;ROUTINE ADDRESS)
 8489   2038    C8  .   .          RZ              ;NO - RETURN
 8490   2039    .   .   .   CRA060 EQU $
 8491   2039    CD  23  20         CALL CRADV1  ;RESET CURADV FLAG
 8492   203C    CD  A6  12         CALL DCXB2D     ;DATA COMM OR I/O BUFF CHAR?
 8493   203F    21  C2  C1         LXI  H,ENDPR*256+XMONLY  ;(SET DEFAULT)
 8494   2042    C2  49  20         JNZ  CRA070     ;YES - DON'T SOUND BELL
 8495   2045    CD  14  48         CALL ZBELL      ;NO - SOUND BELL
 8496   2048    6C  .   .          MOV  L,H        ;LOOK FOR "ENDPR" ONLY
 8497   2049    .   .   .   CRA070 EQU  $
 8498   2049    CD  A0  1E         CALL FNDCH      ;NEXT CHARACTER UNPROTECTED
 8499   204C    CC  C4  1D         CZ   FLDSR        ;OR ANOTHER FIELD EXIST?
 8500   204F    C0  .   .          RNZ             ;YES - RETURN
 8501   2050    CD  A6  12         CALL DCXB2D     ;DATA FROM DATA COMM OR CTU?
 8502   2053    C8  .   .          RZ              ;NO, FROM KEYBOARD - RETURN
 8503   2054    C3  2C  1D         JMP  CURPH1     ;YES - HOME THE CURSOR
 8505   2057   .    .    .    ;*************************
 8506   2057   .    .    .    ; CRADV - ADVANCE CURSOR *
 8507   2057   .    .    .    ;*************************
 8508   2057   .    .    .    CRADV  EQU $
 8509   2057   3A   BE   FF          LDA  RHTMGN     ;GET RIGHT MARGIN SETTING
 8510   205A   21   C1   FF          LXI  H,CURCOL
 8511   205D   .    .    .    CRA010 EQU  $
 8512   205D   BE   .    .           CMP  M          ;CURSOR AT RIGHT MARGIN?
 8513   205E   CA   77   20          JZ   CRA100     ;YES - CHECK FOR WRAP AROUND
 8514   2061   3E   4F   .           MVI  A,MAXCOL     ;(SET FOR LAST COL CHECK)
 8515   2063   FA   5D   20          JM   CRA010     ;AFTER MARGIN - CHECK EOL
 8516   2066   34   .    .           INR  M          ;ADVANCE CURSOR
 8517   2067   BE   .    .           CMP  M          ;MOVED INTO RIGHT MARGIN OR
 8518   2068   C4   65   10          CNZ  CKPROT       ;INTO PROTECTED FIELD?
 8519   206B   C8   .    .           RZ              ;YES - DON'T SET CURADV FLAG
 8520   206C   3A   F4   FF          LDA  MDFLG1     ;GET TERMINAL MODE FLAGS
 8521   206F   E6   02   .           ANI  INSCHR     ;IN CHARACTER INSERT MODE?
 8522   2071   C0   .    .           RNZ             ;YES - DON'T SET FLAG
 8523   2072   2E   67   .           MVI  L,CRAFLG-BASE  ;NO - SET CURADV FLAG
 8524   2074   36   01   .           MVI M,1
 8525   2076   C9   .    .           RET
 8526   2077   .    .    .    ;*************************************
 8527   2077   .    .    .    ; CURSOR IS IN LAST COLUMN OF LINE *
 8528   2077   .    .    .    ;*************************************
 8529   2077   .    .    .    CRA100 EQU  $
 8530   2077   3A   C5   FF          LDA  LSTFMT     ;SAVE LAST FORMAT CONTROL
 8531   207A   32   8A   FF          STA  FMTCTL       ;IN CURRENT LINE
 8532   207D   CD   76   19          CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE OR
 8533   2080   CC   47   10          CZ   CKDSPF       ;DISPLAY FUNCTIONS ENABLED
 8534   2083   C2   96   20          JNZ  CRLF       ;YES - DON'T CLEAR WRAP FLAG
 8535   2086   3A   FB   FF          LDA  KBJMPR     ;NO - GET KEYBOARD JUMPERS 1
 8536   2089   E6   04   .           ANI  LINWRP     ;WRAP AROUND ENABLED?
 8537   208B   C0   .    .           RNZ             ;NO - RETURN
 8538   208C   3A   6E   FF          LDA  DFLGS      ;YES - GET DATA TRANSFER FLG
 8539   208F   E6   80   .           ANI  XBF2DS     ;I/O BUFFER TO DISPLAY?
 8540   2091   3E   BF   .           MVI  A,377Q-WRPFLG  ;(SET CLEAR MASK)
 8541   2093   C4   AA   04          CNZ  CLRMF2     ;YES - CLEAR LINE WRAP FLAG
 8542   2096   .    .    .    ;******************************************
 8543   2096   .    .    .    ; CURSOR SHOULD BE WRAPPED INTO NEXT LINE *
 8544   2096   .    .    .    ; GENERATE CR,LF                          *
 8545   2096   .    .    .    ;******************************************
 8546   2096   .    .    .    CRLF   EQU $
 8547   2096   CD   B8   21          CALL CRRET   ;CARRIAGE RETURN
 8548   2099   C3   6F   0A          JMP LNFEED   ;LINE FEED
 8550   209C   .    .    .   ;********************************
 8551   209C   .    .    .   ; CURPR - CURSOR POINTER RIGHT *
 8552   209C   .    .    .   ;********************************
 8553   209C   .    .    .   CURPR  EQU $
 8554   209C   3E   01   .          MVI A,1       ;GET INCREMENT RIGHT
 8555   209E   C3   A3   20         JMP CURPL1
 8556   20A1   .    .    .   ;*******************************
 8557   20A1   .    .    .   ; CURPL - CURSOR POINTER LEFT *
 8558   20A1   .    .    .   ;*******************************
 8559   20A1   .    .    .   CURPL  EQU $
 8560   20A1   3E   FF   .          MVI A,-1      ;GET INCREMENT LEFT
 8561   20A3   .    .    .   CURPL1 EQU $
 8562   20A3   2E   C1   .          MVI L,CURCOL ;GET CURSOR COLUMN
 8563   20A5   86   .    .          ADD M         ;ADD INCREMENT
 8564   20A6   77   .    .          MOV M,A       ;STORE NEW COLUMN ADDRESS
 8565   20A7   FA   B3   20         JM  CURPL2    ;WRAPAROUND TO LEFT
 8566   20AA   D6   50   .          SUI MAXCOL+1 ;WRAPAROUND TO RIGHT?
 8567   20AC   C0   .    .          RNZ           ;NO - RETURN
 8568   20AD   77   .    .          MOV M,A       ;YES - SET TO COLUMN ZERO
 8569   20AE   .    .    .   ;*******************************
 8570   20AE   .    .    .   ; CURPD - CURSOR POINTER DOWN *
 8571   20AE   .    .    .   ;*******************************
 8572   20AE   .    .    .   CURPD  EQU $
 8573   20AE   3E   01   .          MVI A,1
 8574   20B0   C3   B7   20         JMP CURPU1
 8575   20B3   .    .    .   ;***********************************
 8576   20B3   .    .    .   ; CURSOR MOVED OFF LEFT OF SCREEN *
 8577   20B3   .    .    .   ; WRAPAROUND TO RIGHT AND UP      *
 8578   20B3   .    .    .   ;***********************************
 8579   20B3   .    .    .   CURPL2 EQU $
 8580   20B3   36   4F   .          MVI M,MAXCOL ;PUT CURSOR AT LAST COLUMN
 8581   20B5   .    .    .   ;*****************************
 8582   20B5   .    .    .   ; CURPU - CURSOR POINTER UP *
 8583   20B5   .    .    .   ;*****************************
 8584   20B5   .    .    .   CURPU  EQU $
 8585   20B5   3E   17   .          MVI A,MAXROW
 8586   20B7   .    .    .   CURPU1 EQU $
 8587   20B7   2E   C0   .          MVI L,CURROW ;GET CURSOR ROW
 8588   20B9   86   .    .          ADD M         ;ADD DISPLACEMENT
 8589   20BA   77   .    .          MOV M,A       ;STORE NEW ROW ADDRESS
 8590   20BB   D6   18   .          SUI MAXROW+1 ;ROW LIMIT EXCEEDED?
 8591   20BD   F8   .    .          RM            ;NO - RETURN
 8592   20BE   77   .    .          MOV M,A       ;YES - STORE ADJUSTED ROW
 8593   20BF   C9   .    .          RET           ;RETURN
```

```
8595   20C0   .   .   .    ;****************************
8596   20C0   .   .   .    ; DFSFKY - DEFINE SOFT KEYS *
8597   20C0   .   .   .    ;****************************
8598   20C0   .   .   .    DFSFKY EQU  $
8599   20C0   2E  D9  .           MVI  L,SCRNRW  ;CLEAR SOFT KEY PARAMETERS
8600   20C2   1E  03  .           MVI  E,3         ;TO ZERO
8601   20C4   CD  FF  10          CALL CLRAL1
8602   20C7   21  D0  27          LXI  H,DFSTAB  ;SET RANGE TABLE FOR SOFT KE
8603   20CA   C3  7F  04          JMP  ESCAPA      ;DEFINITION ESCAPE SEQUENC
8604   20CD   .   .   .    ;
8605   20CD   .   .   .    ;  A - DEFINE ATTRIBUTE CODE
8606   20CD   .   .   .    ;
8607   20CD   .   .   .    ;     0 = NORMAL
8608   20CD   .   .   .    ;     1 = LOCAL ONLY
8609   20CD   .   .   .    ;     2 = TRANSMIT ONLY
8610   20CD   .   .   .    ;
8611   20CD   .   .   .    DFS100 EQU  $
8612   20CD   0E  02  .           MVI  C,2       ;SET MAXIMUM VALUE AND
8613   20CF   11  DA  FF          LXI  D,PARM2     ;PARAMETER TO BE SET
8614   20D2   C3  F2  20          JMP  DFS220    ;SET PARAMETER AND EXIT
8615   20D5   .   .   .    ;
8616   20D5   .   .   .    ;  K - KEY NUMBER TO BE DEFINED
8617   20D5   .   .   .    ;
8618   20D5   .   .   .    DFS110 EQU  $
8619   20D5   0E  07  .           MVI  C,NMFCTK-1  ;SET MAXIMUM VALUE AND
8620   20D7   11  D9  FF          LXI  D,PARM3     ;PARAMETER TO BE SET
8621   20DA   C3  E2  20          JMP  DFS200    ;SET PARAMETER AND EXIT
8622   20DD   .   .   .    ;
8623   20DD   .   .   .    ;  L - SET LENGTH OF INPUT
8624   20DD   .   .   .    ;
8625   20DD   .   .   .    DFS120 EQU  $
8626   20DD   0E  4F  .           MVI  C,MAXCOL  ;SET MAXIMUM VALUE AND
8627   20DF   11  DB  FF          LXI  D,PARM1
8628   20E2   .   .   .    ;                     FALL INTO EVALUATION ROUTINE
```

```
8630   20E2   .    .    .   ;
8631   20E2   .    .    .   ;  EVALUATE AND SET PARAMETER
8632   20E2   .    .    .   ;
8633   20E2   .    .    .   ;    D = MAXIMUM ALLOWABLE VALUE
8634   20E2   .    .    .   ;    E = LSB OF PARAMETER TO BE SET (MSB = BASEH)
8635   20E2   .    .    .   ;
8636   20E2   .    .    .   DFS200 EQU  $            ;ENTRY FOR MIN VALUE = 1
8637   20E2   2A   DE   FF         LHLD IODATA       ;GET INPUT PARAMETER
8638   20E5   2B   .    .          DCX  H            ;ADJUST PARAMETER TO ONE LES
8639   20E6   7C   .    .          MOV  A,H          ;CHECK FOR ZERO PARAMETER
8640   20E7   BD   .    .          CMP  L            ;DOES MSB=LSB?
8641   20E8   C2   EF   20         JNZ  DFS210       ;NO - STORE ADJUST VALUE
8642   20EB   3C   .    .          INR  A            ;IS ADJUST VALUE -1
8643   20EC   CA   F2   20         JZ   DFS220       ;YES - DON'T STORE NEW VALUE
8644   20EF   .    .    .   DFS210 EQU  $            ;NO - STORE ADJUSTED VALUE
8645   20EF   22   DE   FF         SHLD IODATA
8646   20F2   .    .    .   DFS220 EQU  $
8647   20F2   CD   11   10         CALL CHKLIO       ;EVALUATE AND SET PARAMETERS
8648   20F5   3A   88   FF         LDA  CHAR         ;RECALL INPUT CHARACTER
8649   20F8   E6   20   .          ANI  40Q          ;IS IT UPPER CASE?
8650   20FA   C2   87   04         JNZ  ESCAPB       ;NO - CONTINUE ESCAPE SEQ
8651   20FD   .    .    .   ;                         YES - SET NEW DEFINITION
```

```
8653   20FD    .    .    .    ;**************************************************
8654   20FD    .    .    .    ; UPPER CASE CHARACTER INPUT - EVALUATE SEQUENCE *
8655   20FD    .    .    .    ;**************************************************
8656   20FD    CD   8C   19           CALL CHKSFK     ;SOFT KEY DEFINE MODE?
8657   2100    CC   69   21           CZ   SWAP       ;NO - SET TO SOFT KEY DISPLA
8658   2103    3A   D9   FF           LDA  PARM3      ;COMPUTE DESIRED KEY DATA RO
8659   2106    87   .    .            ADD  A
8660   2107    3C   .    .            INR  A            ;= 2*(KEY NUMBER) + 1
8661   2108    32   C0   FF           STA  CURROW
8662   210B    21   A6   FF           LXI  H,SFTKYS   ;LOCATE THE START OF THE
8663   210E    CD   F6   0A           CALL MLKSC1       ;DATA ROW
8664   2111    3E   50   .            MVI  A,MAXCOL+1
8665   2113    CD   58   1F           CALL FNDLS0     ;LOCATE THE END OF THE DATA
8666   2116    3E   51   .            MVI  A,MAXCOL+2  ;ROW + 1
8667   2118    90   .    .            SUB  B
8668   2119    32   D9   FF           STA  PARM3      ;SAVE END COLUMN NUMBER
8669   211C    3A   DB   FF           LDA  PARM1      ;TRY TO EXTEND LINE TO
8670   211F    32   C1   FF           STA  CURCOL       ;END OF NEW DATA LINE
8671   2122    CD   9F   22           CALL DSPASC     ;TRY TO ALLOCATE LINE NEEDED
8672   2125    B7   .    .            ORA  A          ;COLUMN POSITION ALLOCATED?
8673   2126    CA   5A   21           JZ   DFS250     ;NO - DON'T SET NEW VALUE
8674   2129    2A   C3   FF           LHLD CURADR     ;YES - GET ADDRESS OF
8675   212C    CD   86   0B           CALL NXTCH0       ;END OF NEW DATA LINE
8676   212F    3A   C1   FF           LDA  CURCOL     ;GET NUMBER OF DATA CHARS
8677   2132    FE   4F   .            CPI  MAXCOL     ;FULL LINE USED?
8678   2134    C4   54   1C           CNZ  CLERLA     ;NO - CLEAR EXCESS CHARACTER
8679   2137    CD   C5   21           CALL CURPRT     ;SET CURRENT COLUMN TO ZERO
8680   213A    21   C0   FF           LXI  H,CURROW   ;SET FOR ATTRIBUTE ROW
8681   213D    35   .    .            DCR  M
8682   213E    3A   DA   FF           LDA  PARM2      ;GET ATTRIBUTE PARAMETER
8683   2141    3D   .    .            DCR  A          ;WHICH ATTRIBUTE TO SET?
8684   2142    3E   4E   .            MVI  A,N          ;(N = NORMAL)
8685   2144    FA   4E   21           JM   DFS230     ;0 - SET AS NORMAL KEY
8686   2147    3E   4C   .            MVI  A,L          ;(L = LOCAL ONLY)
8687   2149    CA   4E   21           JZ   DFS230     ;1 - SET FOR LOCAL ONLY
8688   214C    3E   54   .            MVI  A,T        ;2 - SET FOR TRANSMIT ONLY
8689   214E    .    .    .    DFS230 EQU  $
8690   214E    CD   14   23           CALL DSPTST     ;STORE ATTRIBUTE LETTER
8691   2151    CD   63   21           CALL SWAP0      ;RESTORE ACTIVE DISPLAY
8692   2154    21   F0   27           LXI  H,DFSTB2   ;SET RANGE TABLE FOR SOFT
8693   2157    C3   7F   04           JMP  ESCAPA       ;KEY DATA ACCUMULATION
```

```
 8695    215A    .    .    .    ;*************************************************
 8696    215A    .    .    .    ; NOT ENOUGH BLOCKS AVAILABLE FOR SOFT KEY DATA *
 8697    215A    .    .    .    ;   RESTORE OLD STATE AND IGNORE DEFINITION     *
 8698    215A    .    .    .    ;*************************************************
 8699    215A    .    .    .    DFS250 EQU   $
 8700    215A    3A   D9   FF          LDA   PARM3      ;RECALL END OF DATA LINE
 8701    215D    32   C1   FF          STA   CURCOL
 8702    2160    CD   3C   1C          CALL  CLEARL     ;CLEAR ANY ADDED CHARACTERS
 8703    2163    .    .    .    ;**********************************************
 8704    2163    .    .    .    ; SWAP - SWAP DISPLAY PARAMETERS BETWEEN SOFT *
 8705    2163    .    .    .    ;   KEY AND NORMAL DISPLAY                    *
 8706    2163    .    .    .    ;**********************************************
 8707    2163    .    .    .    ;
 8708    2163    .    .    .    ;  ENTRY:  DON'T CARE
 8709    2163    .    .    .    ;
 8710    2163    .    .    .    ;  EXIT :  DISPLAY PARAMTERS EXCHANGED
 8711    2163    .    .    .    ;          ALL REGISTERS DESTROYED
 8712    2163    .    .    .    ;
 8713    2163    .    .    .    SWAP0  EQU   $
 8714    2163    3A   F8   FF          LDA   CMFLGS     ;GET COMMON FLAGS
 8715    2166    E6   08   .           ANI   DEFSKY     ;DEFINE SOFT KEY MODE?
 8716    2168    C0   .    .           RNZ              ;NO - DON'T DO SWAP
 8717    2169    .    .    .    ;
 8718    2169    .    .    .    SWAP   EQU   $
 8719    2169    21   AE   FF          LXI   H,DSPTYP   ;SET DISPLAY TYPE FLAG
 8720    216C    7E   .    .           MOV   A,M          ;TO VALUE FOR DISPLAY TO
 8721    216D    2F   .    .           CMA                ;MADE ACTIVE
 8722    216E    77   .    .           MOV   M,A
 8723    216F    .    .    .    SWAP1  EQU   $
 8724    216F    0E   0F   .           MVI   C,NUMSWP   ;SET SWAP COUNT
 8725    2171    11   AF   FF          LXI   D,SWPSTR   ;SET ADDRESS OF LOCATIONS
 8726    2174    21   BE   FF          LXI   H,RHTMGN     ;TO BE EXCHANGED
 8727    2177    .    .    .    ;
 8728    2177    .    .    .    ;  EXCHANGE DISPLAY PARAMETERS
 8729    2177    .    .    .    ;
 8730    2177    .    .    .    SWP010 EQU   $
 8731    2177    46   .    .           MOV   B,M        ;GET CURRENT SETTING
 8732    2178    1A   .    .           LDAX  D          ;GET STORED SETTING
 8733    2179    EB   .    .           XCHG             ;EXCHANGE ADDRESSES
 8734    217A    70   .    .           MOV   M,B        ;STORE NEW SAVE VALUE
 8735    217B    12   .    .           STAX  D          ;STORE NEW CURRENT VALUE
 8736    217C    EB   .    .           XCHG             ;RESTORE ADDRESSES
 8737    217D    13   .    .           INX   D          ;INCREMENT TO NEXT VALUE
 8738    217E    23   .    .           INX   H
 8739    217F    0D   .    .           DCR   C          ;ALL VALUES EXCHANGED?
 8740    2180    C2   77   21          JNZ   SWP010     ;NO - MOVE NEXT VALUE
 8741    2183    C9   .    .           RET              ;YES - RETURN
```

```
2184   .   .   .   ;********************
2184   .   .   .   ; SET SOFT KEY DATA *
2184   .   .   .   ;********************
2184   .   .   .   DFS300 EQU  $
2184   CD  8C  19         CALL CHKSFK    ;SOFT KEY ALREADY ENABLED?
2187   CC  69  21         CZ   SWAP      ;NO - SET SOFT KEY DISPLAY O
218A   21  F4  FF         LXI  H,MDFLG1  ;GET SOFT MODE FLAGS
218D   7E  .   .          MOV  A,M
218E   F5  .   .          PUSH PSW       ;SAVE SOFT MODE FLAGS
218F   36  00  .          MVI  M,0       ;FORCE INSERT CHARACTER OFF
2191   CD  B6  14         CALL FDESC1    ;ADD INPUT TO DEFINITION
2194   F1  .   .          POP  PSW       ;RECALL SOFT MODE FLAGS
2195   32  F4  FF         STA  MDFLG1    ;RESTORE ORIGINAL VALUES
2198   CD  63  21         CALL SWAP0     ;RESTORE ACTIVE DISPLAY
219B   CD  30  05         CALL GETDC1    ;SET DISPLAY CURSOR
219E   21  DB  FF         LXI  H,NEWCOL
21A1   35  .   .          DCR  M         ;ALL CHARACTERS DONE?
21A2   F2  8F  04         JP   ESCAP1    ;NO - CONTINUE ESC SEQUENCE
21A5   21  54  26         LXI  H,DFSTB3  ;YES - SET TO WAIT FOR ANY
21A8   C3  7F  04         JMP  ESCAPA      ;CHAR EXCEPT CR, LF, OR DC
21AB   .   .   .   ;********************************************
21AB   .   .   .   ; WAIT FOR CHARACTER TO RESTORE NORMAL MODE *
21AB   .   .   .   ;********************************************
21AB   .   .   .   DFS350 EQU  $             ;LINE FEED CODE
21AB   CD  A6  12         CALL DCXB2D    ;DATA FROM KEYBOARD?
21AE   CA  6F  0A         JZ   LNFEED    ;YES - DO LINE FEED
21B1   C9  .   .          RET            ;NO - RETURN TO RE-ENABLE AL
21B2   .   .   .   ;                       CODES BY CALL TO "ESCEND"
21B2   .   .   .   ;                       IN "CHINT" CLEAN-UP
21B2   .   .   .   ;
21B2   .   .   .   DFS360 EQU  $             ;RETURN CODE
21B2   CD  A6  12         CALL DCXB2D    ;DATA FROM KEYBOARD
21B5   C2  8F  04         JNZ  ESCAP1    ;NO - CONTINUE WAITING
21B8   .   .   .   ;                      YES - DO RETURN OPERATION
21B8   .   .   .   ;************************************
21B8   .   .   .   ; CRRET - SET CURSOR TO LEFT MARGIN *
21B8   .   .   .   ;************************************
21B8   .   .   .   ;
21B8   .   .   .   ;  ENTRY:  DON'T CARE
21B8   .   .   .   ;
21B8   .   .   .   ;  EXIT :  A,CURCOL = LEFT MARGIN SETTING
21B8   .   .   .   ;          IF SPOW NOT DISABLED, SPOW SET
21B8   .   .   .   ;
21B8   .   .   .   CRRET  EQU  $
21B8   3A  FB  FF         LDA  KBJMPR    ;GET STRAP SETTINGS
21BB   E6  02  .          ANI  SPLDIS    ;SPOW DISABLED?
21BD   CA  C5  21         JZ   CURPRT    ;YES - RETURN CURSOR ONLY
21C0   21  6C  FF         LXI  H,SPOWL   ;NO - SET SPOW LATCH
21C3   36  20  .          MVI  M,SPOWON
21C5   .   .   .   CURPRT EQU  $
21C5   3A  BF  FF         LDA  LFTMGN    ;SET CURSOR TO LEFT MARGIN
21C8   .   .   .   CRRET1 EQU  $
```

```
 8795    21C8    32  C1   FF           STA  CURCOL     ;UPDATE CURRENT COLUMN NUMBE
 8796    21CB    32  00   87           STA  IOCRCL       ;AND SET DISPLAY CURSOR
 8797    21CE    C9  .    .            RET             ;RETURN
```

```
=========================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=========================================================================
8799    21CF   .    .    .   ;**********************
8800    21CF   .    .    .   ; DISPLAY ENHANCEMENT *
8801    21CF   .    .    .   ;**********************
8802    21CF   .    .    .   DISPEN EQU  $
8803    21CF   CD   8C   19         CALL CHKSFK     ;DEFINE SOFT KEY MODE?
8804    21D2   C0   .    .          RNZ             ;YES - NO DISPLAY ENHANCEMEN
8805    21D3   21   54   27         LXI  H,DENTAB   ;SET FOR DISPLAY ENHANCEMENT
8806    21D6   C3   81   04         JMP ESCAP0
8807    21D9   .    .    .   ;******************************************
8808    21D9   .    .    .   ; DISPLC - ENTER DISPLAY ENHANCEMENT CHAR *
8809    21D9   .    .    .   ;******************************************
8810    21D9   .    .    .   DISPLC EQU $
8811    21D9   3A   89   FF         LDA  DCHAR      ;GET DISPLAY CHARACTER
8812    21DC   E6   0F   .          ANI  17Q        ;EXTRACT ENHANCEMENT BITS
8813    21DE   .    .    .   DISPC0 EQU  $
8814    21DE   06   30   .          MVI  B,60Q      ;SET MASK TO SAVE ALT CHAR
8815    21E0   .    .    .   ;***********************************************
8816    21E0   .    .    .   ; DISPC1 - ENTER ENHANCEMENT OR FLAG CHARACTER *
8817    21E0   .    .    .   ;***********************************************
8818    21E0   .    .    .   ;
8819    21E0   .    .    .   ;  ENTRY:  A = CHARACTER TO BE STORED
8820    21E0   .    .    .   ;          B = MASK TO SAVE UNCHANGED PART (USED
8821    21E0   .    .    .   ;            ONLY FOR ENHANCEMENT CHARACTERS)
8822    21E0   .    .    .   ;
8823    21E0   .    .    .   ;  EXIT :  SEE "DISPLA"
8824    21E0   .    .    .   ;
8825    21E0   .    .    .   DISPC1 EQU  $
8826    21E0   F6   80   .          ORI  200Q       ;ADD BIT FOR REFRESH LOGIC
8827    21E2   .    .    .   DISPC2 EQU  $
8828    21E2   32   89   FF         STA  DCHAR      ;STORE NEW ENHANCEMENT CODE
8829    21E5   78   .    .          MOV  A,B        ;STORE MASK FOR ENHANCEMENT
8830    21E6   32   77   FF         STA  CDSPEN       ;BITS NOT TO BE ALTERED
8831    21E9   .    .    .   ;                      FALL INTO DISPLAY ROUTINE
```

```
21E9   .  .  .   ;************************************
21E9   .  .  .   ; DISPLA - ADD CHARACTER TO DISPLAY *
21E9   .  .  .   ;************************************
21E9   .  .  .   ;
21E9   .  .  .   ;  ENTRY:  CURCOL,CURROW = SCREEN POSITION WHERE
21E9   .  .  .   ;            CHARACTER IS TO BE INSERTED
21E9   .  .  .   ;          DCHAR = CHARACTER TO BE DISPLAYED
21E9   .  .  .   ;          CDSPEN = MASK TO MASK OUT COMMON BITS
21E9   .  .  .   ;            IF DCHAR IS A DISPLAY CONTROL BYTE
21E9   .  .  .   ;
21E9   .  .  .   ;  EXIT :  A = 0, NO PLACE FOR CHARACTER
21E9   .  .  .   ;          A # 0, CHARACTER PROCESSED
21E9   .  .  .   ;            B = CHARACTER REPLACED IF ADDITION
21E9   .  .  .   ;              DONE BY INSERT
21E9   .  .  .   ;              D,E = ADDRESS OF CHAR IN DISPLAY
21E9   .  .  .   ;
21E9   .  .  .   DISPLA EQU $
21E9   3A 89 FF         LDA  DCHAR  ;GET CHAR TO BE STORED
21EC   B7 .  .          ORA A       ;IS THIS ASCII CHAR?
21ED   F2 9F 22         JP   DIS060 ;YES - CONTINUE
21F0   .  .  .   ;************************************
21F0   .  .  .   ; CONTROL CODE TO BE ENTERED INTO   *
21F0   .  .  .   ; DATA STREAM - FIND CHAR PRECEDING *
21F0   .  .  .   ; THIS COLUMN                       *
21F0   .  .  .   ;************************************
21F0   3A C1 FF         LDA  CURCOL    ;GET CURRENT COLUMN NUMBER
21F3   3D .  .          DCR  A         ;SET FOR PREVIOUS COLUMN
21F4   CD 0B 07         CALL RCADR0    ;DOES LINE EXIST?
21F7   FA 16 0B         JM   MLOCK1    ;NO - SOUND BELL AND EXIT
21FA   .  .  .   ;                        WITH A-REGISTER = 0
21FA   C2 EC 22         JNZ  DIS100    ;COL BEYOND EOL - EXTEND LIN
21FD   .  .  .   ;************************
21FD   .  .  .   ; PREVIOUS COLUMN FOUND *
21FD   .  .  .   ;************************
21FD   4F .  .          MOV C,A       ;SAVE COLUMN IN C
21FE   0C .  .          INR  C         ;SET C TO NEXT COLUMN NUMBER
21FF   CD 65 10         CALL CKPROT    ;PREVIOUS CHAR PROTECTED?
2202   C2 13 22         JNZ  DIS030    ;NO - CONTINUE
2205   .  .  .   DIS020 EQU  $
2205   1B .  .          DCX  D         ;YES - SET PTR TO NEXT CHAR
2206   21 C2 C1         LXI  H,ENDPR*256+XMONLY
2209   CD A0 1E         CALL FNDCH     ;IS NEXT CHARACTER PROTECTED
220C   CA D4 22         JZ   DIS092    ;YES - LOOK FOR NEXT FIELD
220F   21 C1 FF         LXI  H,CURCOL  ;YES - RECALL COLUMN VALUE
2212   4E .  .          MOV  C,M
```

```
8879   2213   .    .    .    ;**********************************
8880   2213   .    .    .    ; SEARCH FOR PLACE FOR CHARACTER *
8881   2213   .    .    .    ;**********************************
8882   2213   .    .    .    DIS030 EQU $
8883   2213   CD   87   0B          CALL NXTCHR  ;GET NEXT CHAR
8884   2216   47   .    .           MOV  B,A        ;SAVE EXISTING CHAR IN B-REG
8885   2217   21   89   FF          LXI  H,DCHAR
8886   221A   FE   C4   .           CPI  STPFLG     ;NON-DISPLAYING TERMINATOR?
8887   221C   CA   3F   22          JZ   DIS035     ;YES - DELETE IT
8888   221F   FE   CC   .           CPI  EOL        ;EXISTING CHARACTER AN EOL?
8889   2221   7E   .    .           MOV  A,M          ;(GET CHAR TO BE DISPLAYED
8890   2222   CA   87   22          JZ   DIS050     ;YES - ADD CHARACTER TO LINE
8891   2225   FE   C4   .           CPI  STPFLG     ;NON-DISPLAYING TERMINATOR?
8892   2227   CA   D5   1A          JZ   CRI104     ;YES - INSERT TERMINATOR
8893   222A   78   .    .           MOV  A,B        ;NO - RECALL EXISTING CHAR
8894   222B   87   .    .           ADD  A          ;EXISTING CHARACTER ASCII?
8895   222C   7E   .    .           MOV  A,M          ;(GET CHAR TO BE DISPLAYED
8896   222D   D2   87   22          JNC  DIS050     ;YES - INSERT NEW CHARACTER
8897   2230   FA   47   22          JM   DIS040     ;FLAG CHAR - ADD FLAG TO DIS
8898   2233   87   .    .           ADD  A          ;NEW CHAR DISPLAY CONTROL?
8899   2234   FA   13   22          JM   DIS030     ;NO - GO TO NEXT CHARACTER
8900   2237   .    .    .    ;************************************
8901   2237   .    .    .    ; MERGE NEW DISPLAY ENHANCEMENT    *
8902   2237   .    .    .    ; WITH CODE ALREADY IN THIS COLUMN *
8903   2237   .    .    .    ;************************************
8904   2237   3A   77   FF          LDA  CDSPEN     ;GET ENHANCEMENT MASK
8905   223A   A0   .    .           ANA B        ;EXTRACT BITS TO BE SAVED
8906   223B   B6   .    .           ORA  M          ;COMBINE WITH NEW ENHANCEMEN
8907   223C   C3   75   22          JMP  DIS044     ;STORE THE NEW DISPLAY CODE
8908   223F   .    .    .    ;********************************************
8909   223F   .    .    .    ; NON-DISPLAYING TERMINATOR FOUND - DELETE IT *
8910   223F   .    .    .    ;********************************************
8911   223F   .    .    .    DIS035 EQU  $
8912   223F   BE   .    .           CMP  M          ;IS NEW CHAR TERMINATOR ALSO
8913   2240   C8   .    .           RZ              ;YES - RETURN
8914   2241   CD   B8   1A          CALL CHRDL2     ;NO - DELETE THE CHARACTER
8915   2244   C3   13   22          JMP  DIS030     ;CONTINUE SCAN
```

```
8917   2247   .   .   .    ;******************
8918   2247   .   .   .    ; FLAG CHAR FOUND *
8919   2247   .   .   .    ;******************
8920   2247   .   .   .    DIS040 EQU $
8921   2247   B8  .   .           CMP B         ;IS THIS SAME FLAG CHAR?
8922   2248   C8  .   .           RZ              ;YES - RETURN (A # 0)
8923   2249   87  .   .           ADD  A          ;NEW CHARACTER DISPLAY CNTL?
8924   224A   1F  .   .           RAR               ;(RESTORE CHARACTER)
8925   224B   F2  78  22          JP   DIS045     ;YES - CHECK PROTECTED FIELD
8926   224E   FE  C5  .           CPI  ALPHA      ;IS NEW CHAR TYPE DEFINITION
8927   2250   78  .   .           MOV  A,B          ;(RECALL OLD FLAG CHAR)
8928   2251   FA  5C  22          JM   DIS042     ;NO - ADD FIELD DEFINITION
8929   2254   FE  C5  .           CPI  ALPHA      ;IS OLD CHAR TYPE DEFINITION
8930   2256   F2  6B  22          JP   DIS043     ;YES - REPLACE THE CHARACTER
8931   2259   C3  13  22          JMP  DIS030     ;NO - GO TO NEXT CHARACTER
8932   225C   .   .   .    ;**********************************************
8933   225C   .   .   .    ; FIELD DEFINITION CHARACTER TO BE ADDED - PUT *
8934   225C   .   .   .    ;   AHEAD OF TYPE DEFINITION OR AFTER "STPR"   *
8935   225C   .   .   .    ;**********************************************
8936   225C   .   .   .    DIS042 EQU  $
8937   225C   FE  C5  .           CPI  ALPHA      ;IS OLD CHAR TYPE DEFINITION
8938   225E   F2  D5  1A          JP   CRI104     ;YES - INSERT FIELD DEF
8939   2261   3E  C0  .           MVI  A,STPR     ;NO - STORE NEW FIELD DEF
8940   2263   B8  .   .           CMP  B          ;OLD CHAR = START PROTECT?
8941   2264   CA  13  22          JZ   DIS030     ;YES - LOOK TO NEXT CHAR
8942   2267   BE  .   .           CMP  M          ;IS NEW CHAR A STPR?
8943   2268   CA  D5  1A          JZ   CRI104     ;YES - INSERT BEFORE UNPROTC
8944   226B   .   .   .    ;**************************************
8945   226B   .   .   .    ; REPLACE EXISTING DISPLAY CHARACTER *
8946   226B   .   .   .    ;**************************************
8947   226B   .   .   .    DIS043 EQU  $
8948   226B   1A  .   .           LDAX D          ;PUT EXISTING CHARACTER INTO
8949   226C   47  .   .           MOV  B,A          ;B-REG FOR SOFT KEY CHECK
8950   226D   3A  89  FF          LDA  DCHAR      ;GET CHAR TO BE DISPLAYED
8951   2270   21  6C  FF          LXI  H,SPOWL    ;CHECK AGAINST SPOW LATCH
8952   2273   BE  .   .           CMP  M          ;INPUT = SPACE AND SPOW SET?
8953   2274   C8  .   .           RZ              ;YES - RETURN (A # 0)
8954   2275   .   .   .    DIS044 EQU  $
8955   2275   12  .   .           STAX D          ;STORE THE NEW CHARACTER
8956   2276   3C  .   .           INR  A          ;FORCE A # 0
8957   2277   C9  .   .           RET             ;RETURN
```

```
===========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
                                                                PAGE 267
===========================================================================
 8959    2278   .   .   .   ;*******************************
 8960    2278   .   .   .   ; FLAG CHAR FOUND AND         *
 8961    2278   .   .   .   ; DISPLAY CONTROL TO BE ADDED *
 8962    2278   .   .   .   ;*******************************
 8963    2278   .   .   .   DIS045 EQU  $
 8964    2278   78  .   .          MOV  A,B       ;RECALL EXISTING CHARACTER
 8965    2279   FE  C0  .          CPI  STPR      ;BEGINNING A PROTECTED FIELD
 8966    227B   C2  13  22         JNZ  DIS030    ;NO - MOVE TO NEXT CHAR
 8967    227E   CD  76  19         CALL CHKFMS    ;FORMAT MODE?
 8968    2281   CA  13  22         JZ   DIS030    ;NO - ADD CHAR TO DISPLAY
 8969    2284   C3  05  22         JMP  DIS020    ;YES - LOOK FOR NEXT FIELD
 8970    2287   .   .   .   ;*****************************************
 8971    2287   .   .   .   ; ASCII OR EOL FOUND                    *
 8972    2287   .   .   .   ; MERGE NEW DISPLAY CONTROL IF NECESSARY *
 8973    2287   .   .   .   ;*****************************************
 8974    2287   .   .   .   DIS050 EQU  $
 8975    2287   87  .   .          ADD  A         ;NEW CHAR DISPLAY CONTROL?
 8976    2288   FA  95  22         JM   DIS054    ;NO - ADD CHAR TO DISPLAY
 8977    228B   3A  77  FF         LDA  CDSPEN    ;YES - GET MASK
 8978    228E   2E  C6  .          MVI  L,LSTDCD-BASE  ;GET LAST ENHANCEMENT
 8979    2290   A6  .   .          ANA M         ;EXTRACT BITS TO BE SAVED
 8980    2291   2E  89  .          MVI  L,DCHAR-BASE
 8981    2293   B6  .   .          ORA  M         ;COMBINE WITH NEW ENHANCEMEN
 8982    2294   77  .   .          MOV M,A       ;STORE
 8983    2295   .   .   .   DIS054 EQU $
 8984    2295   78  .   .          MOV A,B       ;WAS CHAR ASCII?
 8985    2296   B7  .   .          ORA A
 8986    2297   F2  D5  1A         JP  CRI104    ;YES - DO INSERT
 8987    229A   0E  00  .          MVI C,0       ;NO - ADD SINGLE CHAR
 8988    229C   C3  01  23         JMP DIS110
 8990    229F    .    .    .   ;*****************************************
 8991    229F    .    .    .   ; ENTER ASCII CHARACTER INTO DATA STREAM *
 8992    229F    .    .    .   ;*****************************************
 8993    229F    .    .    .   DSPASC EQU  $
 8994    229F    .    .    .   DIS060 EQU $
 8995    229F    CD   08   07         CALL RCADDR  ;GET MEMORY ADDRESS
 8996    22A2    CA   BE   22         JZ  DIS080   ;CHAR FOUND BY RCADDR
 8997    22A5    .    .    .   DISPLO EQU $
 8998    22A5    FA   B4   22         JM  DIS070   ;RETURN IF LINE NOT BUILT
 8999    22A8    0D   .    .          DCR C
 9000    22A9    C2   EC   22         JNZ DIS100   ;MORE THAN ONE CHAR NEEDED
 9001    22AC    .    .    .   ;********************************
 9002    22AC    .    .    .   ; SINGLE CHARACTER REQUIRED      *
 9003    22AC    .    .    .   ; CHECK FOR LAST COLUMN OF LINE *
 9004    22AC    .    .    .   ;********************************
 9005    22AC    FE   4F   .          CPI MAXCOL   ;COMPARE WITH MAX COLUMN
 9006    22AE    C2   01   23         JNZ DIS110   ;NOT MAXIMUM COLUMN
 9007    22B1    C3   CE   22         JMP DIS090
 9008    22B4    .    .    .   ;********************************
 9009    22B4    .    .    .   ; LINE NOT BUILT                 *
 9010    22B4    .    .    .   ; PERFORM HOMEUP IF FORMAT MODE *
 9011    22B4    .    .    .   ;********************************
 9012    22B4    .    .    .   DIS070 EQU  $
 9013    22B4    CD   76   19         CALL CHKFMS     ;FORMAT MODE?
 9014    22B7    C8   .    .          RZ              ;NO - RETURN (A = 0)
 9015    22B8    CD   14   48         CALL ZBELL      ;YES - SOUND BELL
 9016    22BB    C3   E5   22         JMP  DIS093     ;HOME UP AND TRY AGAIN
```

```
9018    22BE    .   .   .    ;************************
9019    22BE    .   .   .    ; CHARACTER REPLACEMENT *
9020    22BE    .   .   .    ;************************
9021    22BE    .   .   .    DIS080 EQU $
9022    22BE    4F  .   .           MOV C,A         ;SAVE COLUMN IN C
9023    22BF    3A  F4  FF          LDA   MDFLG1      ;GET TERMINAL MODE FLAGS
9024    22C2    E6  02  .           ANI   INSCHR      ;IN CHARACTER INSERT MODE?
9025    22C4    CA  CE  22          JZ    DIS090      ;NO - ADD CHARACTER TO DISPL
9026    22C7    3A  90  FF          LDA   EOLMV       ;YES - GET EOL SHIFTED FLAG
9027    22CA    B7  .   .           ORA   A           ;HAS LINE BEEN EXTENDED?
9028    22CB    CA  CF  1A          JZ    CRI100      ;NO - PERFORM INSERT CHAR
9029    22CE    .   .   .    DIS090 EQU   $
9030    22CE    CD  65  10          CALL CKPROT       ;CURSOR IN PROTECTED FIELD?
9031    22D1    C2  6B  22          JNZ   DIS043      ;NO - STORE THE CHARACTER
9032    22D4    .   .   .    DIS092 EQU   $
9033    22D4    CD  A6  12          CALL DCXB2D       ;DATA COMM OR I/O BUFF CHAR?
9034    22D7    CC  14  48          CZ    ZBELL       ;NO - SOUND THE BELL
9035    22DA    3A  89  FF          LDA   DCHAR       ;GET CHAR TO BE DISPLAYED
9036    22DD    B7  .   .           ORA   A           ;IS IT A CONTROL CHARACTER?
9037    22DE    F8  .   .           RM                ;YES - DON'T TAB (RETURN A#0
9038    22DF    CD  C4  1D          CALL FLDSR      ;NO - TAB TO NEXT FIELD
9039    22E2    C2  E9  21          JNZ DISPLA      ;JUMP IF FIELD FOUND
9040    22E5    .   .   .    DIS093 EQU   $
9041    22E5    CD  2C  1D          CALL CURPH1       ;ANY FIELDS IN DISPLAY?
9042    22E8    C2  E9  21          JNZ   DISPLA      ;YES - ADD CHARACTER TO FIEL
9043    22EB    C9  .   .           RET               ;NO - RETURN (A # 0)
```

```
 9045   22EC   .   .   .   ;**************************************************
 9046   22EC   .   .   .   ; LINE MUST BE EXTENDED TO ACCOMODATE CHARACTER  *
 9047   22EC   .   .   .   ;   - EXTEND TO ONE COLUMN BEFORE DESIRED COLUMN *
 9048   22EC   .   .   .   ;**************************************************
 9049   22EC   .   .   .   ;
 9050   22EC   .   .   .   ;  ENTRY:  C = NUMBER OF CHARACTERS REQUIRED
 9051   22EC   .   .   .   ;
 9052   22EC   .   .   .   DIS100 EQU  $
 9053   22EC   0D  .   .          DCR  C            ;MORE THAN ONE CHAR TO ADD?
 9054   22ED   C2  01  23         JNZ  DIS110       ;NO - ADD MULTIPLE CHARACTER
 9055   22F0   CD  65  10         CALL CKPROT       ;CURSOR IN PROTECTED FIELD?
 9056   22F3   CA  D4  22         JZ   DIS092       ;YES - TAB TO NEXT FIELD
 9057   22F6   21  9B  FF         LXI  H,NCHAR      ;NO - SET "NCHAR" TO STORE
 9058   22F9   36  01  .          MVI  M,1            ;BLANK OVER EOL (I.E.,
 9059   22FB   .   .   .   ;                          MAKE DISPLAY ROUTINE
 9060   22FB   .   .   .   ;                          THINK MORE THAN ONE
 9061   22FB   .   .   .   ;                          CHARACTER BEING ADDED)
 9062   22FB   CD  B7  08         CALL DISPL2       ;EXTEND LINE BY ONE CHARACTE
 9063   22FE   C3  0A  23         JMP  DIS114       ;CHECK MEMORY LOCKED
 9064   2301   .   .   .   ;
 9065   2301   .   .   .   DIS110 EQU  $
 9066   2301   CD  65  10         CALL CKPROT       ;CURSOR IN PROTECTED FIELD?
 9067   2304   CA  D4  22         JZ   DIS092       ;YES - TAB TO NEXT FIELD
 9068   2307   CD  AB  08         CALL DISPL1       ;NO - EXTEND LINE
 9069   230A   .   .   .   DIS114 EQU  $
 9070   230A   B7  .   .          ORA  A            ;MEMORY LOCKED?
 9071   230B   C8  .   .          RZ                ;YES - RETURN FAIL (A = 0)
 9072   230C   3A  9B  FF         LDA  NCHAR        ;GET # OF CHARACTERS ADDED
 9073   230F   3D  .   .          DCR  A            ;SINGLE CHARACTER ADDED?
 9074   2310   F2  E9  21         JP   DISPLA       ;NO - TRY TO STORE AGAIN
 9075   2313   C9  .   .          RET               ;YES - STORE DONE BY DISPLAY
 9076   2314   .   .   .   ;                          (A # 0)
```

```
9078   2314   .   .   .   ;
9079   2314   .   .   .   ; * * * * * * * * * * * * * * * * * * * * * * *
9080   2314   .   .   .   ;
9081   2314   .   .   .   ;  DSPTST - DISPLAY TEST PATTERN
9082   2314   .   .   .   ;
9083   2314   .   .   .   ;       ENTRY:  A = CHARACTER TO BE DISPLAYED
9084   2314   .   .   .   ;
9085   2314   .   .   .   ;       EXIT :  A,B,C,D,E,L DESTROYED
9086   2314   .   .   .   ;
9087   2314   .   .   .   DSPTST EQU $
9088   2314   32  89  FF         STA  DCHAR  ;PUT CHAR IN DISPLAY BUFFER
9089   2317   CD  0F  17         CALL SETDF0    ;SET DATA COMM INPUT FLAG TO
9090   231A   .   .   .   ;                     INHIBIT BELL ON FIELD SKIP
9091   231A   .   .   .   ;
9092   231A   .   .   .   ;  DSPCHR - DISPLAY CHARACTER IN DCHAR
9093   231A   .   .   .   ;
9094   231A   .   .   .   DSPCHR EQU  $
9095   231A   21  05  20         LXI  H,CURADV  ;SET NORMAL EXIT ROUTINE
9096   231D   .   .   .   DSPCH0 EQU  $
9097   231D   E5  .   .          PUSH H         ;SAVE NORMAL EXIT ROUTINE
9098   231E   CD  9F  22         CALL DSPASC    ;ADD ASCII CHAR TO DISPLAY
9099   2321   B7  .   .          ORA  A         ;CHARACTER DISPLAYED?
9100   2322   CA  50  23         JZ   DCH100    ;NO - DON'T MOVE CURSOR
9101   2325   .   .   .   ;                     FALL INTO DISPLAY ROUTINE
```

```
 9103     2325   CD  76  19            CALL CHKFMS      ;FORMAT/SOFT KEY DEFINE MODE
 9104     2328   C8  .   .             RZ               ;NO - DO NORMAL EXIT
 9105     2329   CD  A6  12            CALL DCXB2D      ;DATA COMM OR I/O BUFF CHAR?
 9106     232C   C0  .   .             RNZ              ;NO - DO NORMAL EXIT
 9107     232D   3A  89  FF            LDA  DCHAR       ;GET CHARACTER DISPLAYED
 9108     2330   2A  86  FF            LHLD CHKRTN      ;NO - GET CHECK ROUTINE ADDR
 9109     2333   C7  .   .             RST  ;RSTJMP      IS IT A VALID CHARACTER?
 9110     2334   C8  .   .             RZ               ;YES - DO NORMAL EXIT
 9111     2335   F1  .   .             POP  PSW         ;NO - POP OFF NORMAL EXIT AD
 9112     2336   .   .   .     ;
 9113     2336   .   .   .     ;  FIELD CHECK ERROR - LOCK UP UNTIL BACKSPACE HIT
 9114     2336   .   .   .     ;
 9115     2336   AF  .   .             XRA  A           ;CLEAR OUT INPUT CHARACTER
 9116     2337   32  9C  FF            STA  CHARIN        ;TO KILL FUNCTION KEYS
 9117     233A   .   .   .     DCH010 EQU   $
 9118     233A   CD  14  48            CALL ZBELL       ;SOUND BELL
 9119     233D   .   .   .     DCH020 EQU   $
 9120     233D   CD  86  15            CALL IOCTMN      ;MONITOR THE TAPE DRIVES
 9121     2340   CD  05  48            CALL ZGETKY      ;ANY KEY HIT?
 9122     2343   C2  3D  23            JNZ  DCH020      ;NO - CONTINUE WAITING
 9123     2346   FE  0D  .             CPI  CR          ;IS IT THE RETURN KEY?
 9124     2348   C2  3A  23            JNZ  DCH010      ;NO - SOUND BELL, TRY AGAIN
 9125     234B   3E  09  .             MVI  A,STPRPT    ;YES - STOP RETURN KEY
 9126     234D   C3  08  48            JMP  ZKBCTL        ;FROM REPEATING AND EXIT
 9127     2350   .   .   .     ;******************************************
 9128     2350   .   .   .     ; CHARACTER NOT DISPLAYED - SOUND BELL IF *
 9129     2350   .   .   .     ;   CHARACTER FROM KEYBOARD               *
 9130     2350   .   .   .     ;******************************************
 9131     2350   .   .   .     DCH100 EQU   $
 9132     2350   E1  .   .             POP  H           ;POP OFF NORMAL EXIT ROUTINE
 9133     2351   .   .   .     DSPCH1 EQU   $
 9134     2351   CD  A6  12            CALL DCXB2D      ;INPUT FROM KEYBOARD
 9135     2354   CA  14  48            JZ   ZBELL       ;YES - SOUND BELL AND EXIT
 9136     2357   C9  .   .             RET              ;NO - RETURN ONLY
```

```
9138   2358    .    .    .    ;*******************************************
9139   2358    .    .    .    ; EXPAND - EXPAND DISPLAY CONTROL TO ESCAPE *
9140   2358    .    .    .    ;   SEQUENCE                                *
9141   2358    .    .    .    ;*******************************************
9142   2358    .    .    .    ;
9143   2358    .    .    .    ;  ENTRY:  A,C = DISPLAY CONTROL BYTE
9144   2358    .    .    .    ;
9145   2358    .    .    .    ;  EXIT :  H = BASEH
9146   2358    .    .    .    ;          A,B,L DESTROYED
9147   2358    .    .    .    ;
9148   2358    .    .    .    EXPAND EQU  $
9149   2358    CD   05   26          CALL INITD1     ;INITIALIZE CHAR BUFFER PTRS
9150   235B    87   .    .           ADD  A          ;IS CHAR DISPLAY CONTROL?
9151   235C    1F   .    .           RAR               ;(RESTORE CHARACTER)
9152   235D    FA   AC   23          JM   EXP100     ;NO - EXPAND FORMAT CONTROL
9153   2360    21   76   FF          LXI  H,ENHOUT   ;YES - COMPARE TO PREVIOUS
9154   2363    AE   .    .           XRA  M          ;ANY CHANGES?
9155   2364    C8   .    .           RZ            ;NO - RETURN IMMEDIATELY
9156   2365    E6   0F   .           ANI 17Q       ;CHANGE IN ENHANCEMENT?
9157   2367    CA   7E   23          JZ   EXP010     ;NO - CHECK NEW CHARACTER SE
9158   236A    06   26   .           MVI  B,AMPSND   ;YES - OUTPUT ENHANCEMENT
9159   236C    CD   BA   13          CALL ECOUTB       ;ESCAPE SEQUENCE:
9160   236F    3E   64   .           MVI  A,SMALLD     ;<ESC>-<&>-<LOWER CASE D>
9161   2371    CD   C0   13          CALL A2OUTB
9162   2374    79   .    .           MOV  A,C        ;COMPUTE ENHANCEMENT
9163   2375    E6   0F   .           ANI  17Q          ;PARAMETER (@-O)
9164   2377    F6   40   .           ORI  100Q       ;ADJUST TO ASCII LETTER
9165   2379    CD   C0   13          CALL A2OUTB     ;PUT IT INTO OUTPUT BUFFER
9166   237C    2E   76   .           MVI  L,ENHOUT-BASE  ;CHECK CHARACTER SET
```

```
 9168    237E   .    .    .    ;
 9169    237E   .    .    .    ;  CHECK FOR CHARACTER SET CHANGE
 9170    237E   .    .    .    ;
 9171    237E   .    .    .    EXP010 EQU $
 9172    237E   79   .    .           MOV A,C        ;RECALL CURRENT SETTING
 9173    237F   AE   .    .           XRA M          ;COMPARE TO PREVIOUS VALUE
 9174    2380   E6   30   .           ANI 60Q        ;ANY CHANGE IN CHAR SET?
 9175    2382   71   .    .           MOV M,C        ;(SAVE NEW SETTING)
 9176    2383   C8   .    .           RZ             ;NO - RETURN
 9177    2384   79   .    .           MOV A,C        ;YES - RECALL NEW SETTING
 9178    2385   E6   30   .           ANI 600        ;RETURN TO BASE SET?
 9179    2387   CA   A7   23          JZ  EXP030     ;YES - SEND SHIFT IN (SI)
 9180    238A   2E   75   .           MVI  L,CALTST-BASE  ;IS IT THE SAME
 9181    238C   BE   .    .           CMP  M                   ;ALTERNATE CHAR SET?
 9182    238D   CA   A2   23          JZ  EXP020     ;YES - SEND SHIFT OUT ONLY
 9183    2390   77   .    .           MOV M,A        ;NO - SAVE NEW ALTERNATE
 9184    2391   .    .    .    ;
 9185    2391   .    .    .    ;  GENERATE ESCAPE SEQUENCE FOR ALTERNATE
 9186    2391   .    .    .    ;    CHARACTER SET SPECIFIER
 9187    2391   .    .    .    ;
 9188    2391   06   29   .           MVI  B,ARPARN  ;OUTPUT <ESC>
 9189    2393   CD   BA   13          CALL ECOUTB       ;<RIGHT PARENTHESIS>
 9190    2396   79   .    .           MOV  A,C
 9191    2397   E6   30   .           ANI 60Q        ;COMPUTE ALTERNATE CHARACTER
 9192    2399   0F   .    .           RRC            ;SET PARAMETER
 9193    239A   0F   .    .           RRC
 9194    239B   0F   .    .           RRC
 9195    239C   0F   .    .           RRC
 9196    239D   C6   40   .           ADI 100Q
 9197    239F   CD   C0   13          CALL A2OUTB    ;SEND IT
 9198    23A2   .    .    .    ;
 9199    23A2   .    .    .    EXP020 EQU $
 9200    23A2   3E   0E   .           MVI A,SO       ;SEND SHIFT OUT (SO)
 9201    23A4   C3   C0   13          JMP  A2OUTB       ;AND RETURN
 9202    23A7   .    .    .    ;
 9203    23A7   .    .    .    EXP030 EQU $
 9204    23A7   3E   0F   .           MVI A,SI       ;SEND SHIFT IN
 9205    23A9   C3   C0   13          JMP  A2OUTB       ;AND RETURN
```

```
9207    23AC    .    .    .    ;
9208    23AC    .    .    .    ;  EXPAND ON FORMAT CONTROL
9209    23AC    .    .    .    ;
9210    23AC    .    .    .    EXP100 EQU  $
9211    23AC    FE   C2   .           CPI  XMONLY     ;TRANSMIT ONLY CONTROL?
9212    23AE    06   7B   .           MVI  B,LFTBRC     ;(SET FOR LEFT BRACE)
9213    23B0    CA   BA   13          JZ   ECOUTB     ;YES - OUTPUT AND EXIT
9214    23B3    F2   C2   23          JP   EXP110     ;TYPE DEF - OUTPUT NUMBER
9215    23B6    FE   C1   .           CPI  ENDPR      ;END PROTECT?
9216    23B8    06   5B   .           MVI  B,LFTBKT     ;(SET FOR LEFT BRACKET - [
9217    23BA    CA   BA   13          JZ   ECOUTB     ;YES - OUTPUT AND EXIT
9218    23BD    04   .    .           INR  B          ;NO - ALTER CHAR TO  RIGHT
9219    23BE    04   .    .           INR  B            ;BRACKET AND OUTPUT IT
9220    23BF    C3   BA   13          JMP  ECOUTB
9221    23C2    .    .    .    ;
9222    23C2    .    .    .    ;  TYPE DEFINITION - OUTPUT NUMERIC TERMINATOR
9223    23C2    .    .    .    ;
9224    23C2    .    .    .    EXP110 EQU  $
9225    23C2    FE   C8   .           CPI  SFKYAT     ;IS CODE VALID?
9226    23C4    06   7F   .           MVI  B,ADEL       ;(SET DEL CHAR FOR INVALID
9227    23C6    F2   BF   13          JP   B2OUTB     ;NO - RETURN DEL CHARACTER
9228    23C9    FE   C3   .           CPI  FILL       ;FILL CODE?
9229    23CB    CA   BF   13          JZ   B2OUTB     ;YES - RETURN DEL CHARACTER
9230    23CE    D6   8F   .           SUI  ALPHA-6-ZERO  ;COMPUTE ASCII DIGIT
9231    23D0    47   .    .           MOV  B,A        ;PUT CHARACTER INTO B-REG
9232    23D1    C3   BA   13          JMP  ECOUTB     ;OUTPUT THE ESCAPE SEQUENCE
```

```
==============================================================================
 ITEM     LOC    OBJECT CODE   SOURCE STATEMENTS                         PAGE 276
==============================================================================
 9234    23D4    .    .    .    ;
 9235    23D4    .    .    .    ;  GET DISPLAY DATA
 9236    23D4    .    .    .    ;
 9237    23D4    .    .    .    GDS010 EQU  $
 9238    23D4    CD   8C   19          CALL CHKSFK      ;SOFT KEY MODE?
 9239    23D7    CA   70   24          JZ   GDS050      ;NO - DO NORMAL PROCEDURE
 9240    23DA    3A   C0   FF          LDA  CURROW      ;YES - GET CURSOR ROw
 9241    23DD    FE   10   .           CPI  SFTEND      ;BEYOND SOFT KEY DATA?
 9242    23DF    F2   BB   24          JP   GDS160      ;YES - RETURN END OF DISPLAY
 9243    23E2    0F   .    .           RRC              ;IN ATTRIBUTE ROw?
 9244    23E3    DA   3C   24          JC   GDS030      ;NO - OUTPUT DISPLAY DATA
 9245    23E6    06   26   .           MVI  B,AMPSND    ;YES - START ESCAPE SEQUENCE
 9246    23E8    CD   05   26          CALL INITD1      ;INIT OUTPUT BUFFER POINTERS
 9247    23EB    CD   BA   13          CALL ECOUTB      ;SEND <ESC>-<&>
 9248    23EE    3E   66   .           MVI  A,SMALLF      ;<LOWER CASE F>
 9249    23F0    CD   C0   13          CALL A2OUTB
 9250    23F3    3A   C0   FF          LDA  CURROW
 9251    23F6    0F   .    .           RRC                ;<KEY NUMBER>
 9252    23F7    3C   .    .           INR  A
 9253    23F8    F6   30   .           ORI  ZERO
 9254    23FA    CD   C0   13          CALL A2OUTB
 9255    23FD    3E   6B   .           MVI  A,SMALLK      ;<LOWER CASE K>
 9256    23FF    CD   C0   13          CALL A2OUTB
 9257    2402    2A   C9   FF          LHLD LSTLIN      ;GET ADDRESS OF CURRENT
 9258    2405    7D   .    .           MOV  A,L           ;LINE
 9259    2406    D6   08   .           SUI  ATBLOC      ;COMPUTE LOCATION OF
 9260    2408    6F   .    .           MOV  L,A           ;ATTRIBUTE CODE
 9261    2409    7E   .    .           MOV  A,M         ;GET ATTRIBUTE CODE
 9262    240A    06   30   .           MVI  B,ZERO      ;COMPUTE ATTRIBUTE CODE:
 9263    240C    FE   4E   .           CPI  N
 9264    240E    CA   18   24          JZ   GDS020        ;0 = NORMAL
 9265    2411    04   .    .           INR  B             ;1 = LOCAL ONLY
 9266    2412    FE   4C   .           CPI  L             ;2 = TRANSMIT ONLY
 9267    2414    CA   18   24          JZ   GDS020
 9268    2417    04   .    .           INR  B
 9269    2418    .    .    .    GDS020 EQU  $
 9270    2418    CD   BF   13          CALL B2OUTB      ;OUTPUT ATTRIBUTE CODE
 9271    241B    3E   61   .           MVI  A,SMALLA
 9272    241D    CD   C0   13          CALL A2OUTB      ;OUTPUT <LOWER CASE A>
 9273    2420    3E   10   .           MVI  A,FRSOUT    ;SET FLAG TO INDICATE FIRST
 9274    2422    CD   39   17          CALL SETMF2        ;SOFT KEY DATA OUT
 9275    2425    CD   C4   1D          CALL FLDSR       ;LOCATE THE DATA FIELD
 9276    2428    EB   .    .           XCHG
 9277    2429    22   73   FF          SHLD GETADR      ;SAVE FIRST CHAR ADDRESS
 9278    242C    .    .    .    ;                       RESTART "GETDSP" TO OUTPUT
 9279    242C    .    .    .    ;                         FIRST SOFT KEY CHAR
```

```
=========================================================================
ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
=========================================================================
9281    242C   .    .    .    ;
9282    242C   .    .    .    ; * * * * * * * * * * * * * * * * * * * * * * * *
9283    242C   .    .    .    ;
9284    242C   .    .    .    ;  GETDSP - GET A CHARACTER FROM DISPLAY
9285    242C   .    .    .    ;
9286    242C   .    .    .    ;      ENTRY:  CURADR = ADDR OF DISPLAY BYTE
9287    242C   .    .    .    ;              CURCOL = COLUMN OF NEXT BYTE
9288    242C   .    .    .    ;
9289    242C   .    .    .    ;      EXIT :  NC - CHARACTER FOUND
9290    242C   .    .    .    ;                A = CHARACTER
9291    242C   .    .    .    ;                GETADR,CURCOL UPDATED
9292    242C   .    .    .    ;              C - NO CHARACTER
9293    242C   .    .    .    ;                M - END OF DISPLAY
9294    242C   .    .    .    ;                Z - END OF FIELD
9295    242C   .    .    .    ;                P,NZ - END OF LINE
9296    242C   .    .    .    ;              B-L DESTROYED
9297    242C   .    .    .    ;
9298    242C   .    .    .    GETDSP EQU   $
9299    242C   21   3C   FF          LXI   H,B2DPTR  ;GET EXPANSION BUFFER
9300    242F   7E   .    .           MOV   A,M         ;POINTER
9301    2430   2B   .    .           DCX   H         ;SET ADDRESS TO END POINTER
9302    2431   BE   .    .           CMP   M         ;BUFFER EMPTY?
9303    2432   CA   D4   23          JZ    GDS010    ;YES - GET BYTE FROM DISPLAY
9304    2435   2C   .    .           INR   L         ;NO - INCREMENT POINTER AND
9305    2436   3C   .    .           INR   A           ;STORE IT
9306    2437   77   .    .           MOV   M,A
9307    2438   6F   .    .           MOV   L,A       ;SET BUFFER ADDRESS
9308    2439   7E   .    .           MOV   A,M       ;GET THE DATA BYTE
9309    243A   B7   .    .           ORA   A         ;SET C = FALSE
9310    243B   C9   .    .           RET             ;RETURN CHARACTER FOUND
```

```
243C   .    .    .    ;
243C   .    .    .    ;  GET SOFT KEY DATA
243C   .    .    .    ;
243C   .    .    .    GDS030 EQU  $
243C   21   6F   FF          LXI  H,MFLGS2  ;GET MODE FLAGS
243F   7E   .    .           MOV  A,M       ;MASK OUT FIRST OUTPUT FLAG
2440   E6   EF   .           ANI  377Q-FRSOUT
2442   BE   .    .           CMP  M         ;FIRST DATA?
2443   CA   70   24          JZ   GDS050    ;NO - GET NEXT DATA
2446   77   .    .           MOV  M,A       ;YES - UPDATE FLAG
2447   CD   05   26          CALL INITD1    ;INITIALIZE CHAR BUFFER PTRS
244A   2A   73   FF          LHLD GETADR    ;LOCATE END OF LINE
244D   EB   .    .           XCHG           ;PUT START ADDRESS IN D,E
244E   3E   4F   .           MVI  A,MAXCOL  ;SEARCH TO END OF LINE
2450   CD   58   1F          CALL FNDLS0    ;ANY "EOL" IN DATA LINE?
2453   3E   50   .           MVI  A,MAXCOL+1  ;(SET FOR NO EOL LENGTH=80
2455   FA   59   24          JM   GDS040    ;NO - OUTPUT VALUE MAXCOL+1
2458   90   .    .           SUB  B         ;YES -  COMPUTE EOL LOCATION
2459   .    .    .    GDS040 EQU  $
2459   F5   .    .           PUSH PSW       ;SAVE DATA LENGTH
245A   21   C0   13          LXI  H,A2OUTB  ;SET OUTPUT ROUTINE ADDRESS
245D   CD   23   08          CALL BN2DE1    ;CONVERT AND STORE IN BUFFER
2460   3E   4C   .           MVI  A,L       ;OUTPUT UPPER CASE L
2462   CD   C0   13          CALL A2OUTB
2465   F1   .    .           POP  PSW       ;RECALL DATA LENGTH
2466   3D   .    .           DCR  A         ;DOES DATA EXIST?
2467   3E   20   .           MVI  A,ABLNK     ;(SET TO ADD BLANK)
2469   FC   C0   13          CM   A2OUTB    ;NO - ADD A BLANK TO OUTPUT
246C   C3   2C   24          JMP  GETDSP    ;OUTPUT LENGTH PARAMETER
```

```
==========================================================================
ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                     PAGE 279
==========================================================================
9342    246F    .    .    .    ;
9343    246F    .    .    .    ;  GET NEXT BYTE FROM DISPLAY
9344    246F    .    .    .    ;
9345    246F    .    .    .    GDS045 EQU  $         ;ENTRY TO SKIP TERMINATOR
9346    246F    77   .    .           MOV  M,A       ;UPDATE "DFLGS" TO CLEAR
9347    2470    .    .    .    ;                       SKIP TERMINATOR FLAG
9348    2470    .    .    .    GDS050 EQU  $
9349    2470    2A   73   FF          LHLD GETADR  ;GET CURRENT ADDRESS
9350    2473    AF   .    .           XRA  A
9351    2474    B5   .    .           ORA  L         ;END OF DISPLAY?
9352    2475    CA   B8   24          JZ   GDS150    ;YES - TERMINATE
9353    2478    .    .    .    GDS060 EQU  $
9354    2478    7E   .    .           MOV  A,M
9355    2479    2B   .    .           DCX  H         ;DECREMENT TO NEXT BYTE
9356    247A    22   73   FF          SHLD GETADR    ;UPDATE "GETADR"
9357    247D    B7   .    .           ORA  A         ;IS BYTE ASCII?
9358    247E    F2   92   24          JP   GDS100    ;YES - RETURN CHARACTER
9359    2481    FE   D0   .           CPI  LNKLIM    ;IS IT A LINK?
9360    2483    DA   BF   24          JC   GDS200    ;NO - PROCESS DISPLAY CONTRO
9361    2486    6E   .    .           MOV  L,M       ;YES - SET NEW ADDRESS
9362    2487    67   .    .           MOV  H,A
9363    2488    7D   .    .           MOV  A,L       ;PUT LSB INTO A-REGISTER
9364    2489    2F   .    .           CMA
9365    248A    E6   0F   .           ANI  BLKSM     ;IS IT AN END OF LINE LINK?
9366    248C    CA   78   24          JZ   GDS060    ;NO - CONTINUE THRU CHAIN
9367    248F    C3   44   25          JMP  GDS320    ;YES - CHECK TERMINATON
```

```
================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                           PAGE 280
================================================================================
 9369    2492   .   .   .   ;
 9370    2492   .   .   .   ;  ASCII BYTE FOUND - RETURN CHARACTER FOUND
 9371    2492   .   .   .   ;
 9372    2492   .   .   .   GDS100 EQU  $
 9373    2492   47  .   .          MOV  B,A         ;SAVE THE CHARACTER
 9374    2493   11  C1  FF         LXI  D,CURCOL
 9375    2496   1A  .   .          LDAX D           ;INCREMENT CURSOR COLUMN
 9376    2497   3C  .   .          INR  A             ;POSITION
 9377    2498   12  .   .          STAX D
 9378    2499   32  00  87         STA  LOCRCL      ;UPDATE DISPLAY CURSOR
 9379    249C   3A  04  50         LDA  BLKTRM      ;GET BLOCK TERMINATOR CHAR
 9380    249F   B8  .   .          CMP  B           ;IS CHAR = BLOCK TERMINATOR?
 9381    24A0   3E  F7  .          MVI  A,377Q-SKPTRM  ;(SET CLEAR FLAG)
 9382    24A2   CA  AA  24         JZ   GDS110      ;YES - RETURN TERMINATION
 9383    24A5   CD  01  16         CALL CLRDFL      ;NO - CLEAR SKIP FLAG
 9384    24A8   78  .   .          MOV  A,B         ;RECALL DISPLAY CHARACTER
 9385    24A9   C9  .   .          RET              ;RETURN (NC FROM "CLRDFL")
 9386    24AA   .   .   .   ;********************************************
 9387    24AA   .   .   .   ; BLOCK TERMINATOR - CHECK FOR END OF LINE, *
 9388    24AA   .   .   .   ;   RETURN END OF DISPLAY                   *
 9389    24AA   .   .   .   ;********************************************
 9390    24AA   .   .   .   GDS110 EQU  $
 9391    24AA   21  6E  FF         LXI  H,DFLGS     ;CLEAR SKIP TERMINATOR FLAG
 9392    24AD   A6  .   .          ANA  M
 9393    24AE   BE  .   .          CMP  M           ;WAS SKIP FLAG SET?
 9394    24AF   C2  6F  24         JNZ  GDS045      ;YES - IGNORE TERMINATOR
 9395    24B2   1A  .   .          LDAX D           ;NO - TERMINATE TRANSMISSION
 9396    24B3   FE  50  .          CPI  MAXCOL+1    ;WAS RS IN LAST COLUMN?
 9397    24B5   CC  96  20         CZ   CRLF        ;YES - DO CR,LF
 9398    24B8   .   .   .   ;
 9399    24B8   .   .   .   ;  RETURN END OF DISPLAY
 9400    24B8   .   .   .   ;
 9401    24B8   .   .   .   GDS150 EQU  $
 9402    24B8   CD  20  1E         CALL FLDSRX      ;SET "LSTCOL" TO MAXCOL+1 TO
 9403    24BB   .   .   .   ;                         FORCE LINE RE-SCAN
 9404    24BB   .   .   .   GDS160 EQU  $
 9405    24BB   AF  .   .          XRA  A           ;SET A TO -1
 9406    24BC   3D  .   .          DCR  A
 9407    24BD   37  .   .          STC              ;SET C-FLAG TRUE
 9408    24BE   C9  .   .          RET              ;RETURN
```

```
9410    24BF   .   .   .    ;
9411    24BF   .   .   .    ;  NON-ASCII BYTE PROCESSING
9412    24BF   .   .   .    ;
9413    24BF   .   .   .    GDS200 EQU  $
9414    24BF   FE  CE  .           CPI  EOP        ;END OF DISPLAY?
9415    24C1   CA  B8  24          JZ   GDS150     ;YES - RETURN END OF DISPLAY
9416    24C4   FE  C4  .           CPI  STPFLG     ;NON-DISPLAYING TERMINATOR?
9417    24C6   CA  0C  25          JZ   GDS230     ;YES - RETURN END OF DISPLAY
9418    24C9   FE  CC  .           CPI  EOL        ;END OF LINE?
9419    24CB   CA  22  25          JZ   GDS300     ;YES - RETURN END OF LINE
9420    24CE   FE  C3  .           CPI  FILL       ;FILL BYTE?
9421    24D0   CA  78  24          JZ   GDS060     ;YES - GET NEXT BYTE
9422    24D3   4F  .   .           MOV  C,A        ;NO - SAVE THE BYTE
9423    24D4   CD  76  19          CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
9424    24D7   C2  E7  24          JNZ  GDS210     ;YES - LOOK FOR START PROTEC
9425    24DA   3A  64  FF          LDA  IOFLG2     ;NO - GET I/O FLAGS 2
9426    24DD   E6  20  .           ANI  XDS2BF     ;DISPLAY TO I/O BUFFER?
9427    24DF   79  .   .           MOV  A,C          ;(RECALL DATA BYTE)
9428    24E0   C0  .   .           RNZ             ;YES - RETURN UNEXPANDED BYT
9429    24E1   CD  58  23          CALL EXPAND     ;NO - EXPAND DISPLAY CONTROL
9430    24E4   C3  2C  24          JMP  GETDSP     ;RETURN 1ST EXPANDED CHAR
9431    24E7   .   .   .    ;
9432    24E7   .   .   .    ;  FORMAT MODE - IGNORE ALL DISPLAY CONTROL EXCEPT
9433    24E7   .   .   .    ;     FOR START PROTECT
9434    24E7   .   .   .    ;
9435    24E7   .   .   .    GDS210 EQU  $
9436    24E7   79  .   .           MOV  A,C        ;RECALL THE DATA BYTE
9437    24E8   FE  C0  .           CPI  STPR       ;IS IT START PROTECT?
9438    24EA   C2  78  24          JNZ  GDS060     ;NO - IGNORE THE BYTE
9439    24ED   EB  .   .           XCHG            ;YES - PUT GETADR INTO D,E
9440    24EE   2A  C0  FF          LHLD CURROW     ;SAVE ENDING ROW AND
9441    24F1   3A  A3  FF          LDA  TLINO        ;COLUMN+1 FOR FIELD
9442    24F4   85  .   .           ADD  L
9443    24F5   6F  .   .           MOV  L,A        ;SAVE ABSOLUTE ROW NUMBER
9444    24F6   22  20  FF          SHLD ENDROW
9445    24F9   .   .   .    GDS220 EQU  $
9446    24F9   CD  53  10          CALL GTMOD1     ;PAGE MODE/DISPLAY -> BUFFER
9447    24FC   CA  09  25          JZ   GDS225     ;NO - RETURN END OF FIELD
9448    24FF   CD  B9  1D          CALL FLDSR1     ;ANY MORE FIELDS?
9449    2502   CA  B8  24          JZ   GDS150     ;NO - EXIT END OF DISPLAY
9450    2505   EB  .   .           XCHG            ;YES - STORE NEW GETADR
9451    2506   22  73  FF          SHLD GETADR
9452    2509   .   .   .    GDS225 EQU  $
9453    2509   AF  .   .           XRA  A          ;RETURN END OF FIELD
9454    250A   37  .   .           STC               ;(C = TRUE, A = 0)
9455    250B   C9  .   .           RET
```

```
================================================================================
 ITEM     LOC    OBJECT CODE  SOURCE STATEMENTS
================================================================================
 9457   250C    .   .   .    ;**********************************************
 9458   250C    .   .   .    ; NON-DISPLAYING TERMINATOR FOUND - CHECK FOR *
 9459   250C    .   .   .    ;   AUTO CLEAR OPTION                         *
 9460   250C    .   .   .    ;**********************************************
 9461   250C    .   .   .    GDS230 EQU  $
 9462   250C    3A  FA  FF          LDA  KBJMP2    ;GET JUMPERS SET 2
 9463   250F    E6  02  .           ANI  CLRTRM    ;CLEAR TERMINATOR?
 9464   2511    3A  C1  FF          LDA  CURCOL      ;(SET CURRENT COLUMN)
 9465   2514    4F  .   .           MOV  C,A
 9466   2515    2A  73  FF          LHLD GETADR      ;(SET LOCATION OF
 9467   2518    23  .   .           INX  H              ;TERMINATOR)
 9468   2519    EB  .   .           XCHG             ;(PUT ADDRESS INTO D,E)
 9469   251A    C4  B8  1A          CNZ  CHRDL2    ;YES - CLEAR THE BYTE
 9470   251D    EB  .   .           XCHG
 9471   251E    2B  .   .           DCX  H         ;SET LAST CHARACTER ADDRESS
 9472   251F    C3  B8  24          JMP  GDS150    ;RETURN END OF DISPLAY
```

```
2522   .   .   .   ;
2522   .   .   .   ;  END OF LINE - PAD OUT LINE IF FORMAT MODE
2522   .   .   .   ;
2522   .   .   .   GDS300 EQU  $
2522   CD  76  19         CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
2525   CA  41  25         JZ   GDS310     ;NO - ADVANCE TO NEXT LINE
2528   FA  41  25         JM   GDS310     ;SOFT KEY - SKIP TO NEXT LIN
252B   11  C1  FF         LXI  D,CURCOL   ;FORMAT - BLANK FILL
252E   1A  .   .          LDAX D          ;GET CURRENT CURSOR COLUMN
252F   FE  50  .          CPI  MAXCOL+1   ;LINE COMPLETED?
2531   CA  41  25         JZ   GDS310     ;YES - ADVANCE TO NEXT LINE
2534   3C  .   .          INR  A          ;NO - INCREMENT COLUMN
2535   12  .   .          STAX D            ;NUMBER
2536   32  00  87         STA  IOCRCL     ;UPDATE DISPLAY CURSOR
2539   23  .   .          INX  H          ;RESTORE "GETADR" TO LOCATIO
253A   22  73  FF         SHLD GETADR       ;OF "EOL"
253D   3E  20  .          MVI  A,ABLNK    ;RETURN BLANK
253F   B7  .   .          ORA  A          ;CLEAR C-FLAG
2540   C9  .   .          RET             ;RETURN
2541   .   .   .   ;
2541   .   .   .   GDS310 EQU  $
2541   CD  69  19         CALL CHAIN1     ;GET ADDR OF NEXT LINE LINK
```

```
 9497   2544   .    .    .   ;
 9498   2544   .    .    .   ;  EOL LINK FOUND - DETERMINE TERMINATION TYPE
 9499   2544   .    .    .   ;
 9500   2544   .    .    .   GDS320 EQU  $
 9501   2544   7E   .    .          MOV  A,M        ;GET POINTER TO NEXT LINE
 9502   2545   2B   .    .          DCX  H
 9503   2546   6E   .    .          MOV  L,M
 9504   2547   67   .    .          MOV  H,A
 9505   2548   22   73   FF        SHLD GETADR     ;PUT IT INTO "GETADR"
 9506   254B   AF   .    .          XRA  A          ;PUT CURSOR IN COLUMN ZERO
 9507   254C   32   76   FF        STA  ENHOUT     ;CLEAR LAST ENHANCE OUT FLAG
 9508   254F   CD   C8   21        CALL CRRET1
 9509   2552   CD   76   19        CALL CHKFMS     ;FORMAT/SOFT KEY DEFINE MODE
 9510   2555   CA   8F   25        JZ   GDS360     ;NEITHER - SEND END OF LINE
 9511   2558   FA   89   25        JM   GDS350     ;SOFT KEY - FIND NEXT FIELD
 9512   255B   2A   C0   FF        LHLD CURROW     ;FORMAT - SAVE ENDING ROW AN
 9513   255E   3A   A3   FF        LDA  TLINO        ;COLUMN+1 FOR FIELD
 9514   2561   85   .    .          ADD  L
 9515   2562   6F   .    .          MOV  L,A        ;SAVE ABSOLUTE ROW NUMBER
 9516   2563   22   20   FF        SHLD ENDROW
 9517   2566   CD   6F   0A        CALL LNFEED     ;YES - DO LINE FEED
 9518   2569   CD   9E   0F        CALL DISLN1     ;SET DISPLAY CURSOR ROW
 9519   256C   2A   73   FF        LHLD GETADR     ;RECALL POINTER TO NEXT LINE
 9520   256F   7D   .    .          MOV  A,L        ;GET LSB VALUE
 9521   2570   B7   .    .          ORA  A          ;END OF DISPLAY (LSB = 0)?
 9522   2571   CA   BB   24        JZ   GDS160     ;YES - RETURN END OF DISPLAY
 9523   2574   .    .    .   ;
 9524   2574   .    .    .   ;  FORMAT EOL - CHECK FOR CONTINUATION FIELD
 9525   2574   .    .    .   ;
 9526   2574   EB   .    .          XCHG            ;PUT CURRENT ADDR IN D,E
 9527   2575   CD   84   1E        CALL FLDSR2     ;NEXT LINE CONTINUES FIELD?
 9528   2578   C2   F9   24        JNZ  GDS220     ;NO - RETURN END OF FIELD
 9529   257B   EB   .    .          XCHG            ;YES - CONTINUE PROCESSING
 9530   257C   3A   64   FF        LDA  IOFLG2     ;GET I/O FLAGS 2
 9531   257F   E6   20   .          ANI  XDS2BF     ;DISPLAY TO I/O BUFFER?
 9532   2581   CA   78   24        JZ   GDS060     ;YES - CONTINUE FIELD
 9533   2584   22   73   FF        SHLD GETADR     ;NO - STORE NEW "GETADR"
 9534   2587   37   .    .          STC             ;RETURN END OF LINE
 9535   2588   C9   .    .          RET             ;RETURN NZ, C
```

```
=================================================================================
ITEM     LOC    OBJECT CODE  SOURCE STATEMENTS
=================================================================================
9537    2589    .    .    .   ;**********************************************
9538    2589    .    .    .   ; END OF LINE FOR NON-FORMAT MODE - RETURN END *
9539    2589    .    .    .   ;   OF LINE CODE (C, P, NZ)                    *
9540    2589    .    .    .   ;**********************************************
9541    2589    .    .    .   GDS350 EQU  $          ;SOFT KEY END OF LINE
9542    2589    CD   6F   0A         CALL LNFEED     ;LOCATE THE ATTRIBUTE OF THE
9543    258C    CD   C4   1D         CALL FLDSR        ;NEXT DEFINITION
9544    258F    .    .    .   GDS360 EQU  $          ;NON-FORMAT/SOFT KEY EOL
9545    258F    AF   .    .          XRA  A          ;SET NZ,P
9546    2590    3C   .    .          INR  A
9547    2591    37   .    .          STC             ;SET C-TRUE
9548    2592    C9   .    .          RET             ;RETURN END OF LINE
```

```
 9550    2593     .    .    .    ;*************************************
 9551    2593     .    .    .    ; INITDG - INITIALIZE FOR DISPLAY GET *
 9552    2593     .    .    .    ;*************************************
 9553    2593     .    .    .    ;
 9554    2593     .    .    .    ;  EXIT :  Z - CHARACTER FOUND
 9555    2593     .    .    .    ;             GETADR = ADDRESS OF FIRST CHARACTER
 9556    2593     .    .    .    ;          NZ - NO CHARACTER FOUND
 9557    2593     .    .    .    ;             GETADR UNCHANGED
 9558    2593     .    .    .    ;          ALL REGISTERS DESTROYED
 9559    2593     .    .    .    ;
 9560    2593     .    .    .    ;  DISPLAY GET ROUTINF IS SET TO START
 9561    2593     .    .    .    ;  AT CURRENT CURSOR LOCATION
 9562    2593     .    .    .    ;
 9563    2593     .    .    .    INITDO EQU  $           ;ENTRY FOR DISPLAY TO I/O
 9564    2593    CD   7B   19           CALL CHKFMT      ;FORMAT MODE ENABLED?
 9565    2596    C2   9C   25           JNZ  INITDG      ;YES - DON'T MOVE CURSOR
 9566    2599    32   C1   FF           STA  CURCOL      ;NO - BEGIN AT LINE START
 9567    259C     .    .    .    INITDG EQU  $
 9568    259C    CD   0F   17           CALL SETDFO      ;SET DATA COMM INPUT FLAG TO
 9569    259F     .    .    .    ;                          ENABLE TRANSMIT ONLY DATA
 9570    259F    E6   FB    .           ANI  377Q-NOSEND  ;CLEAR NO DATA FLAG
 9571    25A1    BE    .    .           CMP  M           ;WAS IT SET BEFORE?
 9572    25A2    77    .    .           MOV  M,A           ;(SET NEW VALUE)
 9573    25A3    C0    .    .           RNZ              ;YES - RETURN NO DATA
 9574    25A4    2E   64    .           MVI  L,IOFLG2-BASE  ;CLEAR DISPLAY BOUNDARY
 9575    25A6    7E    .    .           MOV  A,M           ;FLAGS
 9576    25A7    E6   3F    .           ANI  377Q-ENDDSP-DSPBTM
 9577    25A9    77    .    .           MOV  M,A
```

```
9579   25AA   .   .   .   ;
9580   25AA   .   .   .   ;  LOCATE FIRST CHARACTER
9581   25AA   .   .   .   ;
9582   25AA   CD  8C  19         CALL CHKSFK     ;SOFT KEY MODE?
9583   25AD   CA  BE  25         JZ   IDG055     ;NO - LOCATE THE FIRST CHAR
9584   25B0   21  C0  FF         LXI  H,CURROW   ;YES - CHECK CURSOR POSITION
9585   25B3   3E  10  .          MVI  A,SFTEND
9586   25B5   BE  .   .          CMP  M          ;CURSOR BELOW DATA AREA?
9587   25B6   F8  .   .          RM              ;YES - RETURN NO CHARACTER
9588   25B7   3E  FE  .          MVI  A,376Q     ;NO - SET CURSOR ROW TO
9589   25B9   A6  .   .          ANA  M            ;ATTRIBUTE ROW
9590   25BA   77  .   .          MOV  M,A
9591   25BB   AF  .   .          XRA  A          ;SET CURSOR COLUMN TO
9592   25BC   23  .   .          INX  H            ;BEGINNING OF ROW
9593   25BD   77  .   .          MOV  M,A
9594   25BE   .   .   .   ;                      LOCATE ATTRIBUTE
9595   25BE   .   .   .   ;
9596   25BE   .   .   .   IDG055 EQU  $
9597   25BE   3E  01  .          MVI  A,IGNTRM   ;SET TO IGNORE NON-DISPLAYIN
9598   25C0   32  6D  FF         STA  TRMFCT       ;TERMINATORS
9599   25C3   CD  CD  06         CALL RCADR4     ;DISPLAY PRESENT?
9600   25C6   F8  .   .          RM              ;NO - RETURN NO CHARACTER
9601   25C7   CA  D3  25         JZ   IDG060     ;CHARACTER - CHECK PROTECTED
9602   25CA   CD  7B  19         CALL CHKFMT     ;EOL - FORMAT MODE?
9603   25CD   CA  EE  25         JZ   IDG100     ;NO - EXIT WITH EOL
9604   25D0   C3  D7  25         JMP  IDG070     ;YES - CHECK PROTECTED
9605   25D3   .   .   .   ;
9606   25D3   .   .   .   IDG060 EQU  $
9607   25D3   21  C2  FF         LXI  H,PROFLD   ;SET PROTECT STATUS
9608   25D6   70  .   .          MOV  M,B
9609   25D7   .   .   .   IDG070 EQU  $
9610   25D7   CD  65  10         CALL CKPROT     ;CURSOR IN PROTECTED FIELD?
9611   25DA   C2  E8  25         JNZ  IDG090     ;NO - RETURN CHARACTER FOUND
9612   25DD   3E  FF  .          MVI  A,STPXFR   ;SET TERMINATOR FUNCTION TO
9613   25DF   32  6D  FF         STA  TRMFCT       ;TERMINATE TRANSFER
9614   25E2   CD  B9  1D         CALL FLDSR1     ;ANY MORE FIELDS?
9615   25E5   CA  01  0B         JZ   NZEXIT     ;NO - RETURN NO CHARACTER
9616   25E8   .   .   .   IDG090 EQU  $
9617   25E8   21  FF  00         LXI  H,377Q     ;INITIALIZE PREVIOUS FIELD'S
9618   25EB   22  20  FF         SHLD ENDROW       ;ROW AND COLUMN TO ZERO
```

```
 9620   25EE    .    .    .    ;
 9621   25EE    .    .    .    ;  CHARACTER FOUND - RETURN CHARACTER FOUND
 9622   25EE    .    .    .    ;
 9623   25EE    .    .    .    IDG100 EQU  $
 9624   25EE    1A   .    .           LDAX D             ;GET FIRST CHARACTER
 9625   25EF    FE   C4   .           CPI  STPFLG        ;NON-DISPLAYING TERMINATOR?
 9626   25F1    CC   87   0B          CZ   NXTCHR        ;YES - GET THE NEXT CHARACTE
 9627   25F4    EB   .    .           XCHG               ;SAVE ADDRESS OF BYTE
 9628   25F5    22   73   FF          SHLD GETADR
 9629   25F8    3A   2A   48          LDA  ALTOUT        ;SET CURRENT ALTERNATE CHAR
 9630   25FB    32   75   FF          STA  CALTST          ;SET TO DEFAULT VALUE
 9631   25FE    3A   C6   FF          LDA  LSTDCD        ;SET LAST ENHANCEMENT OUT
 9632   2601    32   76   FF          STA  ENHOUT          ;WORD
 9633   2604    BF   .    .           CMP  A             ;SET Z-FLAG TRUE
 9634   2605    .    .    .    INITD1 EQU  $             ;INITIALIZE CHARACTER BUFFER
 9635   2605    21   3C   3C          LXI  H,(B2DBFL-1)*256+B2DBFL-1;POINTERS
 9636   2608    22   3B   FF          SHLD B2DEND
 9637   260B    C9   .    .           RET                ;RETURN
```

```
9639   260C   .    .    .   ;*****************************************************
9640   260C   .    .    .   ; STAT2 - SEND SECONDARY TERMINAL STATUS REQUEST *
9641   260C   .    .    .   ;*****************************************************
9642   260C   .    .    .   STAT2  EQU  $
9643   260C   01   00   04         LXI  B,SSTAT2   ;SET SECONDARY STATUS PENDIN
9644   260F   C3   CA   16         JMP  SBLXF0        ;FLAG
9645   2612   .    .    .   ;**************************************************
9646   2612   .    .    .   ; STA2GO - TRANSMIT SECONDARY TERMINAL STATUS *
9647   2612   .    .    .   ;**************************************************
9648   2612   .    .    .   STA2GO EQU  $
9649   2612   01   FF   FB         LXI  B,-1-SSTAT2
9650   2615   CD   70   10         CALL CLBLXF     ;CLEAR STATUS 2 PENDING FLAG
9651   2618   06   7C   .          MVI  B,VRTBAR   ;SEND <ESC>-<VERTICAL BAR>
9652   261A   CD   BB   17         CALL ESCOUT
9653   261D   21   C1   17         LXI  H,XPUTDC   ;SET OUTPUT ROUTINE ADDRESS
9654   2620   CD   26   26         CALL STA2G1     ;OUTPUT SECONDARY STATUS BIT
9655   2623   C3   1D   12         JMP  SDTERM     ;SEND TERMINATOR AND RETURN
```

```
 9657    2626    .   .   .   ;****************************************
 9658    2626    .   .   .   ; STA2G1 - OUTPUT SECONDARY STATUS BITS *
 9659    2626    .   .   .   ;****************************************
 9660    2626    .   .   .   ;
 9661    2626    .   .   .   ;  ENTRY:  H,L = ADDRESS OF OUTPUT ROUTINE
 9662    2626    .   .   .   ;
 9663    2626    .   .   .   ;  EXIT :  ALL REGISTER DESTROYED
 9664    2626    .   .   .   ;          CNTFAD DESTROYED
 9665    2626    .   .   .   ;
 9666    2626    .   .   .   STA2G1 EQU  $
 9667    2626    22  CE  FF         SHLD CNTFAD     ;SET OUTPUT ROUTINE ADDRESS
 9668    2629    .   .   .   STA2G2 EQU  $
 9669    2629    .   .   .   ;
 9670    2629    .   .   .   ;  SEND NON-DISPLAY RAM SIZE (K)
 9671    2629    .   .   .   ;
 9672    2629    3E  D0  .          MVI  A,(BFSPCE+1)/256
 9673    262B    21  8E  FF         LXI  H,BUFBGN+1  ;COMPUTE NON-DISPLAY RAM
 9674    262E    96  .   .          SUB  M           ;SIZE
 9675    262F    CD  9D  0B         CALL PAROT2     ;SEND NON-DISPLAY RAM SIZE
 9676    2632    .   .   .   ;
 9677    2632    .   .   .   ;  OUTPUT TERMINAL TYPE
 9678    2632    .   .   .   ;
 9679    2632    3A  FD  FF         LDA  TRMTYP     ;GET THE TERMINAL TYPE NUMBE
 9680    2635    CD  9F  0B         CALL PAROUT     ;SEND ONLY LOWER FOUR BITS
 9681    2638    .   .   .   ;
 9682    2638    .   .   .   ;  OUTPUT REMAINING KYBD INTFACE STRAPS
 9683    2638    .   .   .   ;
 9684    2638    2A  F9  FF         LHLD KBJMP3     ;GET JUMPERS J-Z
 9685    263B    7C  .   .          MOV  A,H        ;SEND STRAPS J-K-L-M
 9686    263C    CD  9F  0B         CALL PAROUT
 9687    263F    7C  .   .          MOV  A,H        ;SEND STRAPS N-P-Q-R
 9688    2640    CD  9B  0B         CALL PAROT4
 9689    2643    7D  .   .          MOV  A,L        ;SEND STRAPS S-T-U-V
 9690    2644    CD  9F  0B         CALL PAROUT
 9691    2647    7D  .   .          MOV  A,L        ;SEND STRAPS W-X-Y-Z
 9692    2648    CD  9B  0B         CALL PAROT4
 9693    264B    .   .   .   ;
 9694    264B    .   .   .   ;  OUTPUT MEMORY LOCK STATUS
 9695    264B    .   .   .   ;
 9696    264B    3A  6A  FF         LDA  MLKFLG     ;GET MEMORY LOCK FLAG
 9697    264E    21  F4  FF         LXI  H,MDFLG1   ;COMBINE WITH MODE FLAG
 9698    2651    A6  .   .          ANA  M          ;EXTRACT MEMORY LOCK STATE
 9699    2652    E6  04  .          ANI  MEMLOK
 9700    2654    C3  9F  0B         JMP  PAROUT     ;OUTPUT MEMORY LOCK STATE
 9701    2657    .   .   .   ;                       AND RETURN
```

```
9703   2657   .   .   .   ;**********************************************
9704   2657   .   .   .   ; SOFT KEY DATA DONE TABLE - IGNORE DC3,CR,& LF *
9705   2657   .   .   .   ;**********************************************
9706   2654   .   .   .   DFSTB3 EQU  $-3
9707   2657   0A  0A  .          DB   12Q,12Q    ;LINE FEED
9708   2659   AB  A1  .          DW   DFS350+B15  ;CHECK FOR IGNORE
9709   265B   0D  0D  .          DB   15Q,15Q    ;RETURN
9710   265D   B2  A1  .          DW   DFS360+B15  ;CHECK FOR IGNORE
9711   265F   13  13  .          DB   23Q,23Q    ;DC3
9712   2661   8F  84  .          DW   ESCAP1+B15  ;IGNORE IT
9713   2663   .   .   .   ;**********************************
9714   2663   .   .   .   ; SOFT KEY MODE ENABLED RANGE TABLE *
9715   2663   .   .   .   ;**********************************
9716   2660   .   .   .   DFSTB0 EQU  $-3
9717   2663   20  7F  .          DB   40Q,177Q  ;DISPLAYABLE CHARACTER
9718   2665   BA  8C  .          DW   SFKYDS+B15  ;DISPLAY IN PROPER DISPLAY
9719   2667   .   .   .   ;*********************************
9720   2667   .   .   .   ; NORMAL CHARACTER SET ATTRIBUTES *
9721   2667   .   .   .   ;*********************************
9722   2664   .   .   .   RTABLE EQU  $-3
9723   2667   20  7F  .          DB   40Q,177Q  ;ALPHANUMERICS
9724   2669   1A  A3  .          DW   DSPCHR+B15  ;DISPLAYABLE CHARACTERS
9725   266B   07  0F  .          DB   7Q,17Q     ;BELL,BS,HT,LF,VT,FF,CR,SO,S
9726   266D   77  26  .          DW   RTB010     ;USE FUNCTION TABLE
9727   266F   1B  1B  .          DB  33Q,33Q  ;ESCAPE
9728   2671   61  84  .          DW   ESCAPE+B15  ;USE <ESC> RANGE TABLE
9729   2673   .   .   .   ;
9730   2673   00  7F  .          DB   0Q,177Q    ;ALL OTHER CODES
9731   2675   12  84  .          DW   CHKCTL+B15  ;CHECK FOR BLOCK XFR CHARS
9732   2677   .   .   .   ;
9733   2677   .   .   .   ;  <BELL> THROUGH <SHIFT IN>
9734   2677   .   .   .   ;
9735   2677   .   .   .   RTB010 EQU  $
9736   2677   14  48  .          DW   ZBELL      ;BELL - SOUND KEYBOARD BELL
9737   2679   .   .   .   RTB020 EQU  $          ;<BS> THROUGH <SHIFT IN>
9738   2679   F0  1F  .          DW   BCKSPC     ;BS - BACKSPACE CURSOR
9739   267B   82  1F  .          DW   HTAB    ;HORIZONTAL TAB
9740   267D   6F  0A  .          DW   LNFEED  ;LINE FEED
9741   267F   71  19  .          DW   NOFNCT     ;VT - NO FUNCTION
9742   2681   71  19  .          DW   NOFNCT     ;FF - NO FUNCTION
9743   2683   B8  21  .          DW   CRRET   ;CARRIAGE RETURN
9744   2685   D8  0C  .          DW   SHFTOT  ;SHIFT OUT
9745   2687   EB  0C  .          DW   SHFTIN  ;SHIFT IN
```

```
 9747    2689   .    .    .    ;*******************************************
 9748    2689   .    .    .    ; ESCAPE CHARACTER ATTRIBUTES FOR SOFT KEYS *
 9749    2689   .    .    .    ;*******************************************
 9750    2686   .    .    .    SESCTB EQU   $-3
 9751    2689   29   3E   .           DB    51Q,76Q     ;<)> TO (>)
 9752    268B   95   84   .           Dw    ESCEND+B15  ;ABORT ESCAPE SEQUENCE
 9753    268D   4C   4F   .           DB    114Q,117Q  ;<L> TO <O>
 9754    268F   95   84   .           Dw    ESCEND+B15  ;ABORT ESCAPE SEQUENCE
 9755    2691   53   58   .           DB    123Q,130Q  ;<S> TO <W>
 9756    2693   95   84   .           Dw    ESCEND+B15  ;ABORT ESCAPE SEQUENCE
 9757    2695   .    .    .    ;            *** LOWER CASE CHARACTERS ***
 9758    2695   6C   6D   .           DB    154Q,155Q  ;<L> TO <M>
 9759    2697   95   84   .           Dw    ESCEND+B15  ;ABORT ESCAPE SEQUENCE
 9760    2699   79   7B   .           DB    171Q,173Q  ;<Y> TO <[>
 9761    269B   95   84   .           Dw    ESCEND+B15  ;ABORT ESCAPE SEQUENCE
 9762    269D   .    .    .    ;**************************************
 9763    269D   .    .    .    ; NORMAL ESCAPE CHARACTER ATTRIBUTES *
 9764    269D   .    .    .    ;**************************************
 9765    269A   .    .    .    ESCTAB EQU   $-3
 9766    269D   26   26   .           DB    46Q,46Q     ;<&> - AMPERSAND
 9767    269F   8E   96   .           Dw    PRMSEQ+B15  ;PARAMETERIZED SEQUENCE
 9768    26A1   .    .    .    ;
 9769    26A1   29   29   .           DB    51Q,51Q   ;) - SPECIFY ALT CHAR SET
 9770    26A3   7C   8C   .           Dw    SCHRST+B15
 9771    26A5   31   35   .           DB    61Q,65Q     ;<1> TO <5>
 9772    26A7   C1   26   .           Dw    EI1
 9773    26A9   36   38   .           DB    66Q,70Q     ;<6> TO <8>
 9774    26AB   AB   8F   .           Dw    TYPSET+B15   ;DEFINE FIELD TYPE
 9775    26AD   .    .    .    ;
 9776    26AD   3C   3E   .           DB    74Q,76Q     ;(<) TO (>)
 9777    26AF   CB   26   .           Dw    EI1A          ;USE INDEX TABLE
 9778    26B1   .    .    .    ;
 9779    26B1   40   6D   .           DB    100Q,155Q   ;<@> TO <LOWER CASE M>
 9780    26B3   D1   26   .           Dw    EI2           ;USE INDEX TABLE
 9781    26B5   .    .    .    ;              *** LOWER CASE RANGE ***
 9782    26B5   78   7B   .           DB    170Q,173Q   ;<X> TO <LEFT BRACE>
 9783    26B7   2D   27   .           Dw    EI3           ;USE INDEX TABLE
 9784    26B9   .    .    .    ;
 9785    26B9   7E   7E   .           DB    176Q,176Q   ;<^> (TILDE)
 9786    26BB   0C   A6   .           Dw    STAT2+B15    ;TERMINAL STATUS 2
 9787    26BD   .    .    .    ;
 9788    26BD   00   7F   .           DB    0Q,177Q    ;ALL OTHER CODES
 9789    26BF   95   84   .           Dw    ESCEND+B15  ;ABORT ESCAPE SEQUENCE
```

```
9791   26C1   .    .    .   ;**************************************
9792   26C1   .    .    .   ; INDEX TABLES FOR ESCAPE SEQUENCES *
9793   26C1   .    .    .   ;**************************************
9794   26C1   .    .    .   EI1     EQU   $
9795   26C1   5A   15   .           DW    HTBSET     ;1 - HORIZONTAL TAB SET
9796   26C3   60   15   .           DW    HTBCLR     ;2 - HORIZONTAL TAB CLEAR
9797   26C5   FB   10   .           DW    CLRALL     ;3 - CLEAR ALL TABS
9798   26C7   17   17   .           DW    SETLFT     ;4 - SET LEFT MARGIN
9799   26C9   29   17   .           DW    SETRHT     ;5 - SET RIGHT MARGIN
9800   26CB   .    .    .   ;
9801   26CB   .    .    .   EI1A    EQU   $
9802   26CB   38   15   .           DW    FRNCT1     ;< - SET FOREIGN MODE 1
9803   26CD   95   04   .           DW    ESCEND     ;= - INVALID, ABORT SEQUENCE
9804   26CF   3D   15   .           DW    FRNCT2     ;> - SET FOREIGN MODE 2
9805   26D1   .    .    .   ;
9806   26D1   .    .    .   EI2     EQU   $
9807   26D1   AC   12   .           DW    DELAY0     ;@ - PAUSE FOR 1 SECOND
9808   26D3   B5   20   .           DW    CURPU      ;A - CURSOR POINTER UP
9809   26D5   AE   20   .           DW    CURPD    ;CURSOR POINTER DOWN
9810   26D7   9C   20   .           DW    CURPR    ;CURSOR POINTER RIGHT
9811   26D9   A1   20   .           DW    CURPL      ;D - CURSOR LEFT
9812   26DB   9D   08   .           DW    CLEAR      ;E - FULL TERMINAL RESET
9813   26DD   07   11   .           DW    CURPHD     ;F - HOME DOWN
9814   26DF   C5   21   .           DW    CURPRT     ;G - CURSOR RETURN
9815   26E1   F4   17   .           DW    XMOHME     ;H - HOME TO TRANSMIT-ONLY
9816   26E3   82   1F   .           DW    HTAB     ;CURSOR POINTER TAB
9817   26E5   8F   10   .           DW    CLEARS   ;CLEAR DISPLAY
9818   26E7   3C   1C   .           DW    CLEARL   ;CLEAR LINE
9819   26E9   00   0A   .           DW    LININS   ;LINE INSERT
9820   26EB   B7   09   .           DW    LINDEL     ;M - LINE DELETE
9821   26ED   E6   1F   .           DW    IWRPON     ;N - INSERT W/WRAP AROUND ON
9822   26EF   91   19   .           DW    DELWRP     ;O - DELETE CHAR W/WRAPAROUN
9823   26F1   99   19   .           DW    CHRDEL     ;P - DELETE CHARACTER
9824   26F3   D5   1F   .           DW    ICHON      ;Q - INSERT CHARACTER ON
9825   26F5   DC   1F   .           DW    ICHOFF     ;R - INSERT CHARACTER OFF
9826   26F7   27   0C   .           DW    ROLLUP     ;S - ROLL UP
9827   26F9   C5   0B   .           DW    ROLLDN   ;ROLL DOWN
9828   26FB   2D   0B   .           DW    NEXTPG   ;NEXT PAGE
9829   26FD   A9   0B   .           DW    PREVPG   ;PREVIOUS PAGE
9830   26FF   18   1D   .           DW    FORMON   ;FORMAT MODE ON
9831   2701   02   15   .           DW    FORMOF     ;X - FORMAT MODE OFF
9832   2703   7B   14   .           DW    FDISON     ;Y - DISPLAY FUNCTIONS ON
9833   2705   95   04   .           DW    ESCEND   ;INVALID
9834   2707   9E   16   .           DW    PREND    ;END PROTECT
9835   2709   95   04   .           DW    ESCEND   ;INVALID
9836   270B   94   16   .           DW    PRSTRT   ;START PROTECT
9837   270D   F3   0C   .           DW    STATUS     ;^ - SEND TERMINAL STATUS
9838   270F   3F   17   .           DW    SETTRM     ;_ - STORE NON-DISPLAYING
9839   2711   .    .    .   ;                               TERMINATOR CODE
```

```
 9841   2711    .    .    .   ;
 9842   2711    .    .    .   ;  LOWER CASE RANGE FOR 2 CHARACTER ESC SEQUENCES
 9843   2711    .    .    .   ;
 9844   2711    D1   11   .           DW     RLCRSN  ;@ - SCREEN RELATIVE SENSE
 9845   2713    D9   11   .           DW     CURSEN  ;A - ABSOLUTE CURSOR SENSE
 9846   2715    FA   15   .           DW     KBEN1   ;B - ENABLE KEYBOARD
 9847   2717    07   16   .           DW     KBLOKO    ;C - DISABLE (LOCK) KEYBOARD
 9848   2719    C7   16   .           DW     ENTREN    ;D - SEND DISPLAY TO CPU
 9849   271B    68   15   .           DW     IOBNGO    ;E - FAST BINARY READ
 9850   271D    40   12   .           DW     DISMDM    ;F - DISCONNECT MODEM
 9851   271F    CC   0C   .           DW     SFTRST    ;G - SOFT RESET
 9852   2721    27   1D   .           DW     CURPH     ;H - HOME TO UNPROTECTED
 9853   2723    FF   18   .           DW     BKTAB     ;I - BACK TAB
 9854   2725    A5   0C   .           DW     SFKYON    ;J - TURN ON SOFT KEY MENU
 9855   2727    8D   0C   .           DW     SFKYOF    ;K - RESTORE NORMAL DISPLAY
 9856   2729    CB   0A   .           DW     MLKON     ;L - MEMORY LOCK ON
 9857   272B    C0   0A   .           DW     MLKOFF    ;M - MEMORY LOCK OFF
 9858   272D    .    .    .   ;
 9859   272D    .    .    .   EI3     EQU    $         ;LOWER CASE <X> TO <[>
 9860   272D    99   12   .           DW     DCTEST    ;X - DATA COMM SELF-TEST
 9861   272F    70   14   .           DW     MNMDON    ;Y - MONITOR MODE ON
 9862   2731    7D   0D   .           DW     TEST      ;Z - SELF-TEST
 9863   2733    99   16   .           DW     STRXMO    ;[ - START TRANSMIT-ONLY
```

```
9865   2735   .   .   .   ;*********************************************
9866   2735   .   .   .   ; PRMTAB - TABLE FOR SEQUENCES WITH PARAMETERS *
9867   2735   .   .   .   ;*********************************************
9868   2732   .   .   .   PRMTAB EQU  $-3
9869   2735   61  67  .          DB   141Q,147Q  ;LOWER CASE <A> TO <G>
9870   2737   49  27  .          DW   PRM010      ;USE INDEX TABLE
9871   2739   .   .   .   ;
9872   2739   6B  6B  .          DB   153Q,153Q  ;LOWER CASE <K>
9873   273B   20  C8  .          DW   ZSTLKY+B15  ;GO TO SET KEYS ROUTINE
9874   273D   .   .   .   ;
9875   273D   70  70  .          DB   160Q,160Q  ;LOWER CASE <P>
9876   273F   80  95  .          DW   IOCTGO+B15  ;GO TO I/O CONTROL ROUTINE
9877   2741   .   .   .   ;
9878   2741   73  73  .          DB   163Q,163Q  ;LOWER CASE <S>
9879   2743   1D  C8  .          DW   ZSTJPR+B15  ;GO TO SET JUMPERS POUTINE
9880   2745   00  7F  .          DB   0Q,177Q   ;ALL OTHER CODES
9881   2747   95  84  .          DW   ESCEND+B15  ;ABORT ESCAPE SEQUENCE
9882   2749   .   .   .   ;
9883   2749   .   .   .   PRM010 EQU  $          ;LOWER CASE <A> TO <F>
9884   2749   3A  11  .          DW   CURPOS     ;A - CURSOR POSITIONING
9885   274B   11  16  .          DW   LOADR      ;B - BINARY LOADER
9886   274D   1C  16  .          DW   LOADR1     ;C - LOADER SANS MESSAGE
9887   274F   CF  21  .          DW   DISPEN     ;D - DISPLAY ENHANCEMENT
9888   2751   95  04  .          DW   ESCEND     ;E - INVALID, ABORT SEQUENCE
9889   2753   C0  20  .          DW   DFSFKY     ;F - DEFINE FUNCTION KEYS
9890   2755   4C  17  .          DW   SNDCDE     ;G - SEND ATTENTION/FUNCTION
9891   2757   .   .   .   ;                        CODE
```

```
==========================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS
==========================================================================
 9893    2757   .    .    .  ;**********************************************
 9894    2757   .    .    .  ; DENTAB - DISPLAY ENHANCEMENT ESCAPE TABLE *
 9895    2757   .    .    .  ;**********************************************
 9896    2754   .    .    .  DENTAB EQU  $-3
 9897    2757   40   4F   .          DB   100Q,117Q  ;<@>-<O>
 9898    2759   D9   A1   .          DW   DISPLC+B15  ;TURN ON ENHANCEMENT
 9899    275B   .    .    .  ;******************************************
 9900    275B   .    .    .  ; CHRSTB - ALTERNATE CHARACTER SET TABLE *
 9901    275B   .    .    .  ;******************************************
 9902    2758   .    .    .  CHRSTB EQU  $-3
 9903    275B   40   43   .          DB   100Q,103Q  ;<@> - <C>
 9904    275D   82   8C   .          DW   SCHST1+B15  ;SET ALTERNATE CHAR SET
 9905    275F   .    .    .  ;
 9906    275F   00   7F   .          DB   0Q,177Q   ;ALL OTHER CODES
 9907    2761   95   84   .          DW   ESCEND+B15  ;ABORT ESCAPE SEQUENCE
 9908    2763   .    .    .  ;*********************************************
 9909    2763   .    .    .  ; CRPTAB - CURSOR POSITIONING ESCAPE TABLE *
 9910    2763   .    .    .  ;*********************************************
 9911    2760   .    .    .  CRPTAB EQU  $-3
 9912    2763   2B   2B   .          DB   53Q,53Q   ;<+> - PLUS SIGN
 9913    2765   86   92   .          DW   DCPLUS+B15 ;SET SIGN FLAG TO +1
 9914    2767   2D   2D   .          DB  55Q,55Q  ;NEGATIVE REL. POSITIONING
 9915    2769   8B   92   .          DW   DCMNUS+B15 ;SET SIGN FLAG TO -1
 9916    276B   30   39   .          DB  60Q,71Q  ;VALID PARAMETER DIGITS
 9917    276D   62   92   .          DW   DCNUM+B15 ;ACCUMULATE NUMERICAL VALUE
 9918    276F   .    .    .  ;
 9919    276F   43   43   .          DB   103Q,103Q  ;<C>
 9920    2771   4F   91   .          DW   CURPO1+B15  ;SET COLUMN PARAMETER
 9921    2773   .    .    .  ;
 9922    2773   52   52   .          DB   122Q,122Q  ;<R>
 9923    2775   65   91   .          DW   CURPO3+B15  ;SET ROW PARAMETER
 9924    2777   .    .    .  ;
 9925    2777   59   59   .          DB   131Q,131Q  ;<Y>
 9926    2779   5A   91   .          DW   CURPO2+B15  ;SET SCREEN ROW PARAMETER
 9927    277B   .    .    .  ;
 9928    277B   63   63   .          DB   143Q,143Q  ;<LOWER CASE C>
 9929    277D   4F   91   .          DW   CURPO1+B15  ;SET COLUMN PARAMETER
 9930    277F   .    .    .  ;
 9931    277F   72   72   .          DB   162Q,162Q  ;<LOWER CASE R>
 9932    2781   65   91   .          DW   CURPO3+B15  ;SET ROW PARAMTER .
 9933    2783   .    .    .  ;
 9934    2783   79   79   .          DB   171Q,171Q  ;<LOWER CASE Y>
 9935    2785   5A   91   .          DW   CURPO2+B15  ;SET SCREEN ROW PARAMETER
 9936    2787   .    .    .  ;
 9937    2787   20   20   .          DB  40Q,40Q  ;SPACE - IGNORE
 9938    2789   8F   84   .          DW   ESCAP1+B15
 9939    278B   00   7F   .          DB  0,177Q   ;INVALID
 9940    278D   95   84   .          DW   ESCEND+B15
```

```
9942   278F    .    .    .    ;******************************
9943   278F    .    .    .    ; FUNCTION DISABLE ATTRIBUTES *
9944   278F    .    .    .    ;******************************
9945   278C    .    .    .    FDISTB EQU  $-3
9946   278F   0D   0D    .          DB    15Q,15Q    ;RETURN CODE
9947   2791   F2   8F    .          DW    CARRET+B15
9948   2793   1B   1B    .          DB   33Q,33Q  ;ESCAPE
9949   2795   AA   94    .          DW    FDESC+B15
9950   2797   5A   5A    .          DB   132Q,132Q
9951   2799   88   94    .          DW    FDISOF+B15
9952   279B   00   7F    .          DB    0,177Q     ;ALL OTHER CODES
9953   279D   BA   8C    .          DW    SFKYDS+B15  ;ADD CHARACTER TO DISPLAY
```

```
=================================================================================
 ITEM    LOC    OBJECT CODE  SOURCE STATEMENTS                                   
=================================================================================
 9955   279F    .    .    .  ;***************************************
 9956   279F    .    .    .  ; BINARY LOADER CHARACTER ATTRIBUTES *
 9957   279F    .    .    .  ;***************************************
 9958   279C    .    .    .  LDRTAB EQU   $-3
 9959   279F   41   46    .         DB    101Q,106Q  ;<A> - <F>
 9960   27A1   C7   27    .         DW    LI1          ;USE INDEX TABLE
 9961   27A3    .    .    .  ;
 9962   27A3   61   64    .         DB  141Q,144Q ;LOADER COMMAND
 9963   27A5   C7   27    .         DW    LI1          ;USE INDEX TABLE
 9964   27A7    .    .    .  ;
 9965   27A7   0A   0A    .         DB    12Q,12Q   ;LINE FEED
 9966   27A9   8F   84    .         DW    ESCAP1+B15
 9967   27AB   0D   0D    .         DB  15Q,15Q  ;CR
 9968   27AD   8F   84    .         DW    ESCAP1+B15
 9969   27AF   13   13    .         DB  23Q,23Q  ;DC3
 9970   27B1   8F   84    .         DW    ESCAP1+B15  ;IGNORE
 9971   27B3    .    .    .  ;**********************************************
 9972   27B3    .    .    .  ; SNDCTB - ACCUMULATE ATTENTION/FUNCTION CODE *
 9973   27B3    .    .    .  ;**********************************************
 9974   27B0    .    .    .  SNDCTB EQU   $-3
 9975   27B3   30   37    .         DB    60Q,67Q   ;OCTAL DIGITS
 9976   27B5   62   92    .         DW    DCNUM+B15  ;ACCUMULATE VALUE
 9977   27B7    .    .    .  ;
 9978   27B7   41   41    .         DB    101Q,101Q  ;<A>
 9979   27B9   5A   97    .         DW    SNDCD1+B15  ;SEND ATTENTION CODE
 9980   27BB    .    .    .  ;
 9981   27BB   46   46    .         DB    106Q,106Q  ;<F>
 9982   27BD   CE   92    .         DW    SNDCD2+B15  ;SEND FUNCTION CODE
 9983   27BF    .    .    .  ;
 9984   27BF   20   20    .         DB    40Q,40Q   ;SPACE
 9985   27C1   8F   84    .         DW    ESCAP1+B15
 9986   27C3   00   7F    .         DB  0,177Q   ;OTHER CHARACTERS
 9987   27C5   00   80    .         DW    B15          ;TERMINATE AND RESET
 9988   27C7    .    .    .  ;
 9989   27C7    .    .    .  LI1    EQU   $
 9990   27C7   39   16    .         DW    LDR3       ;A - ADDRESS
 9991   27C9   27   16    .         DW    LDR0     ;B - IGNORE
 9992   27CB   75   16    .         DW    LDR10    ;CHECKSUM
 9993   27CD   4A   16    .         DW    LDR4     ;DATA
 9994   27CF   5A   16    .         DW    LDR060     ;E - EXECUTE LOADED CODE
 9995   27D1   00   00    .         DW    BEGIN      ;F - TERMINATE AND RESET
```

```
===========================================================================
 ITEM     LOC    OBJECT CODE   SOURCE STATEMENTS                        PAGE 299
===========================================================================
 9997     27D3    .    .    .    ;*********************************
 9998     27D3    .    .    .    ; DFSTAB - DEFINE SOFT KEYS TABLE *
 9999     27D3    .    .    .    ;*********************************
10000     27D0    .    .    .    DFSTAB EQU  $-3
10001     27D3    20   20   .           DB    40Q,40Q    ;SPACE
10002     27D5    8F   84   .           Dw    ESCAP1+B15  ;IGNORE
10003     27D7    30   39   .           DB    60Q,71Q    ;DIGITS <0>-<9>
10004     27D9    62   92   .           Dw    DCNUM+B15   ;ACCUMULATE NUMERICAL VALUE
10005     27DB    41   41   .           DB    101Q,101Q  ;<A> - ATTRIBUTE PARAMETER
10006     27DD    CD   A0   .           Dw    DFS100+B15  ;STORE DEFINED ATTRIBUTE
10007     27DF    4B   4C   .           DB    113Q,114Q  ;<K> - <L>
10008     27E1    EF   27   .           Dw    DFT010       ;USE INDEX TABLE
10009     27E3    .    .    .    ;
10010     27E3    .    .    .    ;  LOWER CASE RANGE
10011     27E3    .    .    .    ;
10012     27E3    61   61   .           DB    141Q,141Q  ;<A> - ATTRIBUTE PARAMETER
10013     27E5    CD   A0   .           Dw    DFS100+B15  ;STORE DEFINED ATTRIBUTE
10014     27E7    6B   6C   .           DB    153Q,154Q  ;<K> - <L>
10015     27E9    EF   27   .           Dw    DFT010       ;USE INDEX TABLE
10016     27EB    00   7F   .           DB    0Q,177Q    ;ALL OTHER CODES
10017     27ED    95   84   .           Dw    ESCEND+B15  ;ABORT ESCAPE SEQUENCE
10018     27EF    .    .    .    ;
10019     27EF    .    .    .    DFT010 EQU  $
10020     27EF    D5   20   .           Dw    DFS110      ;DEFINE KEY NUMBER
10021     27F1    DD   20   .           Dw    DFS120      ;DEFINE LENGTH OF INPUT DATA
10022     27F3    .    .    .    ;********************************
10023     27F3    .    .    .    ; ACCUMULATE SOFT KEY DATA TABLE *
 0024     27F3    .    .    .    ;********************************
10025     27F0    .    .    .    DFSTB2 EQU  $-3
10026     27F3    00   7F   .           DB    0Q,177Q    ;ALL CODES
10027     27F5    84   A1   .           Dw    DFS300+B15  ;ADD TO DATA LINE
```

```
==================================================================================
 ITEM    LOC   OBJECT CODE  SOURCE STATEMENTS                            PAGE 300
==================================================================================
10029   27F7   .    .    .         END
    0  ERRORS FOUND IN ASSEMBLY CODE .
```

```
SYMBOL  VALUE  REFERENCED ON
========================================================================
A        0041     317
A2OUTB   13C0    5490, 5485, 7413, 7432, 9161, 9165, 9197, 9201, 9205, 9249,
                 9254, 9256, 9272, 9332, 9335, 9339
ABCKSL   005C     332, 3978
ABLNK    0020     299, 2669, 2889, 2916, 2982, 3090, 5029, 5750, 6988, 7003,
                 7083, 7642, 9338, 9490
ABSTAK   FF5F     743,  747
ADEL     007F     347, 1915, 5879, 9226
ALCC     0063     338, 5097
ALPHA    00C5     358, 4531, 7071, 7097, 7292, 8203, 8926, 8929, 8937, 9230
ALPHNM   00C7     360
ALTIN    0040     850, 1120
ALTIO    0010     768
ALTORG   6000     275,  276
ALTOUT   482A     205, 9629
AMPSND   0026     300, 1395, 5087, 9158, 9245
ANL      0080     760
ANR      0040     759
ARPARN   0029     302, 1400, 9188
ATB010   14E0    5753, 5756
ATBLEN   000E    5757, 1209, 1217
ATBLIN   14DA    5749, 5757, 1200
ATBLOC   0008    5756, 5589, 9259
ATSIGN   0040     316, 6021
AUTOLF   0004     124, 1436, 4018, 5453, 5618, 6327
AUTTRM   0001      56, 5336, 6451
B15      8000     368, 9708, 9710, 9712, 9718, 9724, 9728, 9731, 9752, 9754,
                 9756, 9759, 9761, 9767, 9770, 9774, 9786, 9789, 9873, 9876,
                 9879, 9881, 9898, 9904, 9907, 9913, 9915, 9917, 9920, 9923,
                 9926, 9929, 9932, 9935, 9938, 9940, 9947, 9949, 9951, 9953,
                 9966, 9968, 9970, 9976, 9979, 9982, 9985, 9987,10002,10004,
                10006,10013,10017,10027
B1LEN    FF38     812,  813
B1STAT   FF3A     810,  811
B1TYPE   FF39     811,  812
B2D050   0845    2740, 2713
B2D100   0858    2765, 2737, 2739, 2742, 2744
B2D110   085B    2768, 2771
B2D200   086A    2786, 2706, 2733
B2D210   0875    2793, 2788
B2D220   0876    2795, 2790
B2DBFL   003D     794, 7481, 9635, 9635
B2DBUF   FF3D     793,  794,  795
B2DEND   FF3B     796,  810, 5491, 7479, 9636
B2DPTR   FF3C     795,  796, 9299
B2LEN    FF35     815,  819
B2OUTB   13BF    5486, 9227, 9229, 9270
B2STAT   FF37     813,  814
B2TYPE   FF36     814,  815
BACKT0   1802    6601, 6764
BACKT1   1805    6603, 6479
BACKT5   18E8    6742, 3125
BASE     FF00     506,  794, 1494, 2338, 2599, 2605, 2611, 3313, 3765, 4886,
                 4955, 4966, 4975, 6153, 6364, 6375, 6679, 6767, 6803, 6816,
                 6968, 7017, 7250, 7341, 7898, 8323, 8423, 8523, 8978, 8980,
SYMBOL  VALUE  REFERENCED ON
=========================================================================
               9166, 9180, 9574
BASE2    FE00   508, 1217
BASEH    00FF   505,  506,  507, 1616, 2571
BASEH2   00FE   507,  508
BCKSPC   1FF0  8422, 9738
BEGIN    0000   904,  914,  922,  930,  937,  945,  954,  962, 2647,  906,
               9995
BELLIM   0008   377, 1425
BFSPCE   CFFF   578, 1069, 9672
BINOCT   0802  2668, 4424, 4426
BINXMT   0002   382
BKT010   1922  6785, 6778
BKT050   1926  6796, 6781
BKT060   1938  6806, 6800
BKT100   1939  6811, 6770
BKT110   1947  6827, 6857
BKT120   1949  6829, 6854
BKT130   194D  6835, 6852
BKT150   195B  6850, 6831
BKT210   1830  6620, 6670
BKT220   1839  6628, 6661
BKT230   183B  6630, 6615
BKT240   1868  6652, 6649
BKT250   186B  6654, 6682
BKT300   187A  6667, 6619
BKT310   1886  6677, 6640
BKT400   1890  6687, 6636
BKT410   18AC  6703, 6719
BKT420   18B5  6708, 6715
BKT430   18D1  6724, 6710
BKT450   18DB  6736, 6699, 6720
BKT500   18F0  6750, 6643, 6651, 6659
BKT510   18F6  6753, 6728
BKTAB    18FF  6762, 9853
BLKFIL   FF91   575,  576, 2321, 2338, 3051, 7232
BLKMDE   0002   123, 1712, 2132, 4018, 4724, 5331
BLKSM    000F   378, 1140, 1951, 1957, 1980, 1997, 2037, 2893, 2923, 3026,
               3058, 3625, 5830, 6872, 7361, 7519, 7575, 7593, 7612, 9365
BLKSZ    0010   379, 1137, 1156, 1178, 2000, 2900, 3201
BLKTRG   0001    80, 1665
BLKTRM   5004   233,  234, 1646, 5441, 5646, 6321, 6476, 9379
BN2DE0   081D  2704, 1003
BN2DE1   0823  2707, 5096, 9333
BN2DE2   0826  2709, 5119
BN2DEC   082E  2731, 1002, 4431
BN0010   080C  2676, 2687
BNRYGO   2829   888,  889, 1683
BOT      0020   727
BRKDC    123B  5160, 5710, 6554
BSYCHK   283A   897,  898, 5919
BUFBGN   FF8D   580,  581, 1073, 1827, 3575, 9673
BUFBSY   0080   770
BUFEND   FF8B   581,  585, 1070, 1826, 1830, 3577
BUFMSG   0F27  4443,  999, 1844
C        0043   318, 1405
```

```
SYMBOL  VALUE  REFERENCED ON
=========================================================================
CALTST   FF75    611,  612, 9180, 9630
CAPSLK   0001    122, 4018
CAR010   100B   4614, 4600
CARRET   0FF2   4598, 9947
CDSPEN   FF77    609,  610, 8830, 8904, 8977
CHAIN    196D   6874, 1356, 1689, 1965, 2467, 2476, 3431, 3714, 5592, 5822,
                5834, 5947, 6657, 7363
CHAIN0   1968   6868, 7110, 7133, 7940
CHAIN1   1969   6870, 9495
CHAR     FF88    587,  588, 1566, 1633, 4996, 5222, 6128, 8648
CHARIN   FF9C    566,  567, 1333, 1378, 1432, 5612, 5614, 9116
CHD000   1A0A   7021, 6954
CHD010   19A4   6951, 6947
CHD020   19D0   6987, 7009
CHD050   1A01   7013, 6982, 7004
CHD100   1A19   7042, 7027
CHD110   1A22   7052, 7072, 7103
CHD120   1A49   7077, 7057, 7066
CHD130   1A57   7089, 7080
CHD140   1A5E   7095, 7093
CHD150   1A6C   7101, 7098
CHD200   1A70   7109, 7064
CHD210   1A74   7116, 7054
CHD250   1A88   7132, 7062
CHD260   1A8C   7135, 7121, 7125
CHD400   1A91   7147, 7065, 7099, 7117
CHD500   1A9A   7163, 7024
CHD510   1AA1   7170, 7175, 7184
CHD515   1AAE   7176, 7172
CHD520   1AB0   7182, 7174
CHEKCC   0040     73
CHI000   0342   1471, 1463
CHI010   035F   1493, 1556
CHI020   038D   1523, 1503
CHI030   0393   1528, 1526
CHI050   03A8   1549, 1487
CHI100   03B9   1563, 1483, 1499, 1505, 1510, 1551, 1554
CHI110   03C4   1575, 1585, 1590
CHI200   03E5   1612, 1601
CHI270   040A   1632, 1624, 1628
CHINT    0350   1479, 1467, 1475, 1901, 2151, 5613
CHINT0   0330   1459,  989
CHINT1   03B9   1562, 1473, 2150
CHINT2   03A0   1541, 1906
CHK010   1025   4653, 4695
CHK050   1028   4661, 4647
CHK060   1034   4671, 4688
CHK070   1037   4675, 4649, 4689
CHK100   1038   4680
CHK150   103A   4686, 4666
CHK160   1041   4693, 4665
CHKCT1   0424   1656, 5357, 6293
CHKCTL   0412   1645, 9731
CHKFM0   1972   6894, 6763, 8321
CHKFMS   1976   6897, 1417, 2149, 2369, 2489, 3111, 3193, 3362, 3508, 4528,
SYMBOL  VALUE  REFERENCED ON
=========================================================================
               4798, 5361, 5380, 5450, 6240, 6941, 7287, 7455, 7541, 7875,
               8213, 8447, 8532, 8967, 9013, 9103, 9423, 9478, 9509
CHKFMT  197B   6901, 5420, 6361, 6372, 6463, 7026, 7150, 7841, 9564, 9602
CHKLIO  1011   4640, 4994, 8647
CHKLIM  1017   4643,  973
CHKMLK  1981   6916, 2170, 6461
CHKRIN  FF86    588,  589, 3367, 8209, 8240, 8484, 8488, 9108
CHKSFK  198C   6932, 1367, 1716, 1750, 2129, 2211, 3879, 3899, 3922, 3944,
               3961, 4599, 4912, 5333, 5433, 5675, 5695, 6946, 7554, 7776,
               8656, 8747, 8803, 9238, 9582
CHKSUM  0881   2819, 4153, 4199, 4305
CHRDEL  1999   6945, 6957, 9823
CHRDL1  1A19   7041, 6996, 7206, 7426
CHRDL2  1AB8   7203, 4876, 7655, 8253, 8914, 9469
CHRINS  1AC1   7229, 7488
CHRLOC  0002   5758, 1215
CHRSET  FF72    616,  617, 1278, 3866, 3946
CHRSTB  2758   9902, 3854
CHSAV   FF98    570,  572, 6953, 6955, 6989, 7002, 7007, 7046, 7081, 7123,
               7148, 7424, 7427
CIL     0001    698
CIR     0002    697
CKBRKY  000A    218, 6548
CKDSPF  1047   4702, 1550, 1623, 1913, 2123, 3883, 8533
CKEDIT  104D   4709, 2176, 2248, 3336, 3389, 4096, 5501, 5531, 7757
CKIOKY  0008    216, 4098, 5322
CKLNMD  105F   4733, 5373
CKPROT  1065   4742, 2378, 3053, 7247, 7907, 8518, 8869, 9030, 9055, 9066,
               9610
CKRMTE  106A   4749, 5179, 5259, 6301, 6527
CLA010  1100   4900, 4904
CLBLXF  1070   4769,  974, 3977, 5086, 5466, 5635, 9650
CLCMFL  130C   5521, 3882, 6131, 8407
CLEAR   089D   2849, 9812
CLEARL  1C3C   7538,  982, 3129, 4800, 8702, 9818
CLEARS  108F   4795,  983, 9817
CLER01  1C9C   7632, 4859
CLER02  1C9D   7634, 7069, 7313
CLERL0  1C5E   7568, 7440
CLERL1  1C61   7570, 7127, 7137
CLERLA  1C54   7553, 7542, 8678
CLL120  1C7F   7598, 7602
CLL160  1C8A   7608, 7595
CLL310  1C98   7618, 7613
CLL400  1C9A   7625, 7543
CLL510  1C9F   7637, 7645, 7656, 7664, 7669, 7679
CLL540  1CB0   7649, 7641
CLL544  1CB3   7654, 7668
CLL550  1CB9   7661, 7650
CLL580  1CCA   7674, 7639
CLRAL1  10FF   4898, 1063, 1739, 4186, 4292, 8601
CLRALL  10FB   4885, 9797
CLRDFL  1601   6084, 1336, 4797, 5887, 6485, 7771, 9383
CLRMF2  04AA   1767, 4954, 5079, 5543, 8541
CLRTRG  0000    257, 4787
```

```
SYMBOL  VALUE  REFERENCED ON
=======================================================================
CLRTRM   0002      57, 9463
CLRXON   1088    4786, 5130, 6268
CLS100   10C9    4837, 4799
CLS110   10D4    4848, 4841
CLS120   10D9    4852, 4872
CLS130   10DD    4858, 4843
CLS200   10E4    4866, 4870
CLS210   10E5    4868, 4877
CMBASE   00FF     140,  141
CMDEXC   0008     725
CMFLGS   FFF8     149,  150, 1019, 1337, 1664, 1875, 4490, 4750, 5325, 5522,
                 5556, 7446, 8714
CMND     FF55     751,  772, 2611, 5921, 6169
CMPLIM   FF46     792,  793
CMSTOR   FF00     141
CNTFAD   FFCE     513, 1614, 2708, 2734, 3998, 9667
CNTRLC   FF62     731,  741
CNTXFR   0002     644
COMMA    002C     304
COMMON   FFFF     139,  140,  143
CONDIS   0001      27, 2122
CONDLF   0A69    3306, 4331, 4358
CONDTN   2814     878,  879, 5743
COUNT    FF84     593,  594, 2902, 2936
CR       000D     292, 1310, 1433, 1472, 1553, 1896, 5430, 5615, 6324, 9123
CRA010   205D    8511, 8515
CRA040   2028    8480, 8455
CRA060   2039    8490, 8466
CRA070   2049    8497, 8494
CRA100   2077    8529, 8513
CRADV    2057    8508, 8446
CRADV1   2023    8470, 1279, 1567, 2316, 4395, 5467, 5701, 8491
CRAFLG   FF67     667,  689, 1480, 8472, 8523
CRI100   1ACF    7246, 9028
CRI104   1AD5    7249, 8892, 8938, 8943, 8986
CRI110   1AD9    7253, 7277, 7286, 7288, 7293
CRI120   1ADF    7258, 7256
CRI140   1AF1    7274, 7295, 7364, 7400
CRI150   1B22    7300, 7291
CRI152   1B27    7304, 7306
CRI154   1B2F    7309, 7303, 7311
CRI158   1B39    7319, 7281
CRI159   1B44    7326, 7283
CRI160   1B47    7334, 7322
CRI170   1B48    7336, 7325
CRI180   1B49    7338, 7376
CRI200   1B5D    7356, 7279
CRI240   1B6E    7368, 7362
CRI260   1B83    7387, 7378
CRI300   1B89    7397, 7269
CRI305   1B91    7402, 7265
CRI310   1BAF    7420, 7434
CRI320   1BC3    7431, 7412
CRI330   1BD8    7445, 7415
CRI400   1C0C    7478, 7466, 7470, 7490
```

```
SYMBOL  VALUE  REFERENCED ON
=======================================================================
CRI450   1C25   7494, 7473, 7482
CRI500   1C2E   7516, 7305, 7310
CRI510   1C39   7523, 7520
CRLF     2096   8546, 4349, 4387, 4388, 4585, 4591, 4615, 6482, 8328, 8358,
                8379, 8534, 9397
CRP025   117D   4993, 4968, 4977
CRP050   1180   4995, 4984
CRP200   1192   5011, 5006
CRP500   11A6   5035, 5012
CRPTAB   2760   9911, 4957
CRRET    21B8   8786, 4330, 8547, 9743
CRRET1   21C8   8794, 5363, 5382, 7985, 9508
CRS100   1215   5117, 5106
CRSNGO   11E4   5084, 1697
CRTOFF   0080    410, 1193, 2853, 4183, 6172
CSU100   0884   2823, 2827, 2831
CSU110   089A   2841, 2838
CTBDLY   0020    775, 2608
CTBLNK   FF53    773,  774
CTBLTM   FF52    774,  776, 2605
CTDCDP   282C    889,  893, 5913
CTIADR   FF33    819,  820
CTIBPT   FF2F    821,  822
CTICNT   FF2C    822,  823
CTICSM   FF2A    824,  825
CTIJMP   FFE0    164,  165, 1110
CTINTR   283D    898, 6014
CTISPT   FF31    820,  821
CTISTA   FF29    825,  829
CTITRL   FF2B    823,  824
CTIVEC   FFE1    163,  164, 1108
CTLLIM   0020    298, 1413, 1486
CTLRED   2808    874,  875, 5744
CTMON    282F    893,  894, 5936
CTRDKY   00A0   5722
CTSTAT   FF66    689,  700
CTUIN    0080    849, 1127
CUR100   11DE   5080, 5073
CURAD2   2002   8442, 4340, 4375
CURADR   FFC3    530,  532, 1496, 1530, 2095, 2386, 2540, 2559, 3212, 3818,
                4538, 6745, 7988, 8040, 8481, 8674
CURADV   2005   8445, 1533, 4385, 8443, 9095
CURCOL   FFC1    538,  539, 1422, 1543, 2329, 2331, 2362, 2402, 2443, 2945,
                3068, 4951, 4966, 5021, 5095, 5766, 6227, 6364, 6374, 6696,
                6765, 6961, 7015, 7341, 7349, 7450, 7486, 7898, 8323, 8423,
                8453, 8510, 8562, 8670, 8676, 8701, 8795, 8858, 8876, 9374,
                9464, 9481, 9566
CURFKY   FFA4    554,  555, 5593, 5873, 5878
CURPD    20AE   8572, 9809
CURPH    1D27   7769,  978, 1231, 9852
CURPH1   1D2C   7774, 6576, 8392, 8503, 9041
CURPHD   1107   4911,  979, 6621, 9813
CURPL    20A1   8559, 9811
CURPL1   20A3   8561, 8555
CURPL2   20B3   8579, 8565
```

```
SYMBOL  VALUE  REFERENCED ON
=====================================================================
CURPO1   114F   4963, 9920, 9929
CURPO2   115A   4972, 9926, 9935
CURPO3   1165   4981, 9923, 9932
CURPO4   1198   5020, 6769, 6805, 6844, 8361
CURPOS   113A   4950, 9884
CURPR    209C   8553, 9810
CURPRT   21C5   8792, 3130, 3225, 3507, 4914, 7775, 8679, 8789, 9814
CURPU    20B5   8584, 9808
CURPU1   20B7   8586, 8574
CURROW   FFC0    539,  540, 1194, 1511, 2232, 2237, 2446, 2561, 2986, 3083,
                3228, 3313, 3386, 3506, 3738, 3746, 4514, 4943, 4975, 4989,
                5007, 5042, 5104, 6119, 6494, 6497, 6607, 6616, 6623, 6638,
                6672, 6673, 6711, 6716, 6725, 6737, 6755, 6776, 6948, 6990,
                6997, 7458, 7495, 7547, 7847, 7856, 7868, 7998, 8033, 8034,
                8587, 8661, 8680, 9240, 9250, 9440, 9512, 9584
CURSEN   11D9   5077, 9845
D        0044    319, 8043
DATATR   0040    739
DATCOM   0020    769
DBLHOL   0010    726
DC2      0012    295
DC2GO    1228   5137, 1693
DC2SND   0080     46, 5355, 6290
DC3      0013    296, 1474
DCC010   124C   5185, 5180
DCERR    1251   5190, 6168
DCH010   233A   9117, 9124
DCH020   233D   9119, 9122
DCH100   2350   9131, 9100
DCHAR    FF89    586,  587, 2948, 2980, 3006, 3071, 4337, 4530, 5028, 7230,
                7250, 7284, 7339, 7398, 8811, 8828, 8850, 8885, 8950, 8980,
                9035, 9068, 9107
DCIOFF   0010    104
DCJMP0   0080     61
DCJMP1   0001     65
DCJMP2   0002     66
DCJMP3   0004     67
DCJMP4   0008     68
DCJMS2   5006    235
DCJMSK   5005    234,  235
DCM010   128D   5247, 5241
DCMCT1   124D   5187, 1038, 5545
DCMCTL   1242   5177, 1658, 4788, 5132, 5143, 5162, 5315, 5511, 5657, 6427
DCMERR   0001     89, 3984
DCMINT   1234   5151,  936
DCMNUS   128B   5245,  971, 9915
DCN005   126C   5221, 5219
DCN010   127B   5230, 5233
DCNUM    1262   5215,  969, 9917, 9976,10004
DCPLUS   1286   5239,  970, 9913
DCTEST   1299   5258, 1000, 5324, 9860
DCXB2D   12A6   5274, 3924, 4611, 5677, 5697, 8131, 8492, 8501, 8767, 8774,
                9033, 9105, 9134
DECRDX   000A    130, 1732
DEFSKY   0008     83, 3881, 3901, 4491, 8715
```

```
SYMBOL  VALUE  REFERENCED ON
==========================================================================
DELAY    12B3   5290, 1439
DELAY0   12AC   5282, 9807
DELTRM   0000    655, 2555, 8105
DELWRP   1991   6940, 9822
DENTAB   2754   9896, 8805
DEVFLG   FE7F    847,  854, 1131, 2626, 6012
DFCTOF   149D   5685, 5711
DFLGS    FF6E    641,  652, 1034, 1300, 1415, 1649, 1708, 4111, 4434, 5275,
                5636, 6071, 6085, 6345, 6414, 8538, 9391
DFS100   20CD   8611,10006,10013
DFS110   20D5   8618,10020
DFS120   20DD   8625,10021
DFS200   20E2   8636, 8621
DFS210   20EF   8644, 8641
DFS220   20F2   8646, 8614, 8643
DFS230   214E   8689, 8685, 8687
DFS250   215A   8699, 8673
DFS300   2184   8746,10027
DFS350   21AB   8766, 9708
DFS360   21B2   8773, 9710
DFSFKY   20C0   8598, 9889
DFSTAB   27D0  10000, 8602
DFSTB0   2660   9716, 1752
DFSIB2   27F0  10025, 8692
DFSTB3   2654   9706, 8761
DFT010   27EF  10019,10008,10015
DIS020   2205   8871, 8969
DIS030   2213   8882, 8870, 8899, 8915, 8931, 8941, 8966, 8968
DIS035   223F   8911, 8887
DIS040   2247   8920, 8897
DIS042   225C   8936, 8928
DIS043   226B   8947, 8930, 9031
DIS044   2275   8954, 8907
DIS045   2278   8963, 8925
DIS050   2287   8974, 8890, 8896
DIS054   2295   8983, 8976
DIS060   229F   8994, 8852
DIS070   22B4   9012, 8998
DIS080   22BE   9021, 8996
DIS090   22CE   9029, 9007, 9025
DIS092   22D4   9032, 7248, 8875, 9056, 9067
DIS093   22E5   9040, 9016
DIS100   22EC   9052, 8863, 9000
DIS110   2301   9065, 8988, 9006, 9054
DIS114   230A   9069, 9063
DIS120   08CF   2898, 2901
DIS140   08E1   2915, 2929
DIS160   08FA   2935, 2904
DIS170   0915   2955, 2947, 2961
DIS175   0916   2957, 2950
DIS180   091D   2965, 3016, 3030
DIS210   0933   2983, 2981
DIS220   094A   3002, 2881
DIS240   0954   3013, 2920
DIS400   095A   3021, 2888
```

```
SYMBOL  VALUE  REFERENCED ON
===================================================================
 ISCNT   0006    263, 5167
 ISLN1   0F9E   4513, 1529, 1905, 2995, 3095, 5435, 7751, 8044, 9518
DISLN2   0FA1   4515, 5284
DISLN3   0FA4   4517, 1622
DISLN4   0FA5   4519, 5201, 6166
DISLNK   0F95   4506, 3171, 3290, 7835
DISMDM   1240   5166, 9850
DISPC0   21DE   8813, 4369, 4374
DISPC1   21E0   8825, 3956, 4532, 6244
DISPC2   21E2   8827, 4579, 6404, 7006
DISPEN   21CF   8802, 9887
DISPL0   22A5   8997, 5030
DISPL1   08AB   2877, 7347, 9068
DISPL2   08B7   2884, 9062
DISPLA   21E9   8849, 7234, 9039, 9042, 9074
DISPLC   21D9   8810, 9898
DISPLY   0004    766
DISPST   FFFE    143,  144, 2094, 2553, 4493, 7715, 7750
DLY010   12BC   5296, 5305
DLY020   12C0   5299, 5303
DMAOFF   0060    409, 4507
DOOCTI   2835    895,  896, 1107
DPS100   1308   5345
DPS200   130E   5353, 5332
DPS210   1318   5358
DPS215   131B   5360, 5374
DPS220   1322   5368, 5334, 5337
DPSEN1   1331   5378, 6496
 PSEND   12D8   5321, 5709
 PSGO    133B   5394, 1699
DSG010   1346   5399, 5424, 5436
DSG020   134E   5406, 5410
DSG100   135D   5415, 5408
DSG110   1376   5429, 5421
DSG200   138A   5440, 5396, 5416
DSG210   1399   5449, 5419
DSG220   13A7   5458, 5444, 5451
DSG225   13AA   5460, 5445
DSG230   13B1   5464, 5411
DSM010   1CE3   7705, 7713
DSM500   1CFB   7723, 7699
DSM510   1D01   7726, 7735
DSP010   0437   1671, 1677
DSP020   044D   1688, 1673
DSPASC   229F   8993, 8671, 9098
DSPBGN   FFAA    551,  552, 1081, 1146, 1840, 3580, 4002
DSPBIM   0040    716, 9576
DSPCH0   231D   9096, 4616
DSPCH1   2351   9133, 6366, 6377
DSPCHR   231A   9094, 3923, 3926, 4341, 4612, 5700, 9724
DSPEND   FFA8    552,  553, 1078, 1136, 1839, 1842
DSPFNC   0001    111, 4704, 5667, 5689
DSPLIM   FBFF    491, 1077
DSPMS0   1CD6   7694, 5196, 6121
DSPMS1   1CD7   7696, 1268, 4110, 4113, 5263
```

```
SYMBOL  VALUE  REFERENCED ON
=================================================================
DSPMSG   1CDA   7698,  967
DSPSTR   FE4F    498, 1199, 1209, 1215, 1217, 7700, 7714
DSPTAB   0451   1692, 1702, 1668
DSPTCH   0429   1663, 1326
DSPTST   2314   9087, 4363, 4383, 4561, 8690
DSPTYP   FFAE    544,  550, 1230, 6898, 6933, 8719
ECONTF   FFCD    512,  513,  518,  542, 1014, 1050, 1270, 1620, 2748, 2779,
                3653
ECOUTB   138A   5483, 9159, 9189, 9213, 9217, 9220, 9232, 9247
EDIT     0010    115, 4711
EDTWRP   0008     59
EI1      26C1   9794, 9772
EI1A     26CB   9801, 9777
EI2      26D1   9806, 9780
EI3      272D   9859, 9783
ELM100   0992   3079, 3066, 3070, 3073
ELM110   099E   3089, 3093
ELM130   09AF   3100, 3098
ENDBLK   0007    264, 5131
ENDCOL   FF21    839,  840
ENDDSP   0080    717, 9576
ENDPR    00C1    354, 2397, 4871, 5750, 5754, 6221, 6602, 6602, 7551, 8132,
                8132, 8493, 8873, 9215
ENDROW   FF20    840, 9444, 9516, 9618
ENDTST   0006    214, 2596, 4392
ENHLIM   00BF    352, 7056, 7276
ENHNCF   00FF   5734, 1380
ENHOUT   FF76    610,  611, 9153, 9166, 9507, 9632
ENL100   13D5   5509, 5502
ENR100   13F0   5539, 5532
ENTLCL   13C7   5500, 1923
ENTRCD   0098   5719, 5327
ENTREM   13E2   5530, 1878
ENTREN   16C7   6252, 9848
EOF      0001    733
EOL      00CC    364, 1504, 2944, 3014, 3028, 3064, 4080, 4574, 4938, 5750,
                5752, 7044, 7061, 7136, 7280, 7389, 7557, 8197, 8283, 8283,
                8312, 8888, 9418
EOLADR   FF94    573,  574, 2886, 2985, 7388
EOLMOV   0971   3056, 2879
EOLMV    FF90    576,  577, 3103, 8170, 9026
EOLMV0   0969   3050, 2557
EOP      00CE    365, 2049, 2200, 3491, 4427, 4444, 4461, 4464, 4467, 4470,
                4571, 4810, 4860, 4873, 6044, 7676, 7944, 8199, 8314, 9414
ERREOP   0F50   4460, 4418
ERRFLG   FFF7    150,  151, 3982, 4046, 4379, 6181
ESC      001B    297, 1375, 1555, 5484, 5682, 5750, 6520
ESC010   0473   1715, 1710
ESCAP0   0481   1733, 1718, 3855, 6135, 6419, 8806
ESCAP1   048F   1740, 5235, 5253, 8760, 8775, 9712, 9938, 9966, 9968, 9970,
                9985,10002
ESCAPA   047F   1731, 4958, 6203, 8603, 8693, 8762
ESCAPB   0487   1736, 4998, 8650
ESCAPE   0461   1707, 9728
ESCEN1   04A1   1753, 1751, 5670
```

```
SYMBOL  VALUE  REFERENCED ON
===================================================================
ESCEND   0495   1748,  972, 1276, 1630, 1914, 5251, 5688, 9752, 9754, 9756,
                9759, 9761, 9789, 9803, 9833, 9835, 9881, 9888, 9907, 9940,
               10017
ESCFLG   FFD1    176,  179, 1419, 1625, 1741, 1756
ESCINP   0008    635, 1298, 1713, 1757
ESCLWD   00E4   5727, 1394
ESCOUT   17BB   6519, 3979, 5088, 9652
ESCSO    008E   5726, 1404
ESCTAB   269A   9765, 1717
EVD      0002    734
EW       0080    729
EXP010   237E   9171, 9157
EXP020   23A2   9199, 9182
EXP030   23A7   9203, 9179
EXP100   23AC   9210, 9152
EXP110   23C2   9224, 9214
EXPAND   2358   9148,  993, 9429
EXTB2D   0001    714
F        0046    320, 4041
F1CODE   00F0   5728, 5730, 1363
F8CODE   00F7   5729, 1365
FCR005   1ED6   8174, 8184
FCR010   1EDA   8180, 8191, 8206, 8214, 8222, 8242, 8252, 8254
FCR100   1EEE   8196, 8189
FCR110   1F24   8223, 8220
FCR150   1F26   8229, 8204
FCR160   1F3E   8239, 8232, 8235, 8237
FCR200   1F45   8248, 8202
FCR250   1F55   8258, 8251
FCR260   1F56   8260, 8182
FCR400   1EC8   8142, 8104
FCT200   1432   5608, 5597, 5616, 5619
FCT210   1436   5611, 5621
FCTAD1   01E0   5730, 5731, 5731
FCTADJ   FFDF   5731, 5582
FCTK2D   0010    647, 1301, 1416, 5598, 5637, 5886
FCTKEY   1406   5579, 1368
FDESC    14AA   5694, 9949
FDESC1   14B6   5699, 3927, 5680, 5696, 8753
FDISOF   1488   5674, 9951
FDISON   147B   5664, 9832
FDISTB   278C   9945, 5669
FDO100   147D   5666, 5660
FF       000C    291
FILCHR   FF8F    577,  580, 1944, 2001, 7571, 7603
FILL     00C3    356, 1509, 1941, 2960, 3074, 7063, 7126, 7282, 7439, 7600,
                7663, 9228, 9420
FILNUM   FF5E    747,  748
FILRED   0004    704, 1313
FIVE     0035    312,  943
FKEYGO   1451   5633, 1698
FKG010   145D   5642, 5650
FLDSEP   00C4    363
FLDSR    1DC4   7901, 3509, 4849, 8390, 8499, 9038, 9275, 9543
FLDSR1   1DB9   7895, 7861, 9448, 9614
```

```
SYMBOL  VALUE  REFERENCED ON
=========================================================================
FLDSR2   1E84   8066, 7678, 7963, 8069, 9527
FLDSRB   1E83   8063, 8465
FLDSRX   1E20   7972, 3909, 6405, 9402
FLINE    FF9F    557,  564, 2098, 2246, 2257, 3162, 3300, 7818, 7821, 7872
FLS010   1F61   8290, 8303
FLS020   1F6C   8296, 8293
FLS030   1F6D   8298, 8295
FLS035   1F72   8302, 8315
FLS040   1F75   8304, 8313
FLS050   1F78   8311, 8300
FMICTL   FF8A    585,  586, 8463, 8531
FNCLIM   00A1   5724, 1349
FNCLWR   0098   5723, 1351
FNCTAB   14BC   5708, 1354
FNDCH    1EA0   8098, 2398, 8498, 8874
FNDCH0   1E9D   8096, 2384, 8486
FNDCHR   1ECF   8168, 2554, 8143
FNDCHU   1EB9   8130, 7929
FNDCU1   1EC4   8138, 7915, 8133
FNDLS0   1F58   8281, 7469, 8665, 9326
FNDLST   1F5C   8285, 6635
FNDRAM   04B0   1784, 1072, 1080, 1797
FNDTAB   14E9   5765, 5895, 5903
FNDTB1   14ED   5768, 6820, 8334
FNDTB2   14FA   5788,  984
FOF010   1494   5679, 5676
FORGN    0080    118, 4079
FORMAT   0008    114, 5800, 6903, 7761, 7842
FORMOF   1502   5799, 9831
FORMON   1D18   7756, 9830
FOUR     0034    311,  935
FPS      0004    724
FRBLKS   FFAC    550,  551, 1154, 1945, 1966, 2293, 2295, 4812, 4821, 5812,
                6466
FRC010   150F   5813, 5831
FRC050   151E   5825, 5835
FRC100   1530   5839, 5819
FRCPTY   0080     75
FRCRST   0004     82, 1020, 2851, 5197, 6125, 6130
FRECNT   150A   5810,  980, 3337, 5841
FRM010   04C4   1801, 1789, 1792
FRNCT1   1538   5848, 9802
FRNCT2   153D   5854, 9804
FRNMD1   000E    222, 5849
FRNMD2   000F    223, 5855
FRSALT   4829    204,  205, 1277, 4084
FRSOUT   0010    636, 9273, 9318
FRSIBL   FF92    574,  575
FS2000   1E80   8061, 8073
FS2005   1E83   8064, 8075
FSR080   1DD5   7913, 7966
FSR100   1DE2   7928, 7900, 7908, 7957, 7967
FSR120   1DE8   7936, 7916
FSR140   1E20   7973, 7938, 7945
FSR200   1E26   7982, 7930
```

```
SYMBOL  VALUE  REFERENCED ON
=========================================================================
 SR240   1E42   8000, 8028
 SR300   1E55   8021, 8025
FSR340   1E64   8032, 8003
FSR360   1E7B   8042, 7997
FST      0004    755
FSTBIN   000A    267
FSTRAM   9100     14,  136,  503, 1052, 4208
FSTSND   0020     41
FTB100   14FC   5790, 5793
FULDUP   0080     18, 2139
FWD      0002    754
GAP      0020    693
GBL100   0576   1974, 1958
GBL200   0584   1996, 1969
GBL210   058D   2002, 2006
GDC010   050C   1882, 1902
GDC020   0514   1890, 1916
GDC030   052A   1900, 1894, 1897
GDC050   0536   1911, 1884
GDC100   0544   1921, 1876
GDS010   23D4   9237, 9303
GDS020   2418   9269, 9264, 9267
GDS030   243C   9315, 9244
GDS040   2459   9330, 9328
GDS045   246F   9345, 9394
GDS050   2470   9348, 9239, 9320
GDS060   2478   9353, 9366, 9421, 9438, 9532
GDS100   2492   9372, 9358
 DS110   24AA   9390, 9382
GDS150   24B8   9401, 9352, 9415, 9449, 9472
GDS160   24BB   9404, 9242, 9522
GDS200   24BF   9413, 9360
GDS210   24E7   9435, 9424
GDS220   24F9   9445, 9528
GDS225   2509   9452, 9447
GDS230   250C   9461, 9417
GDS300   2522   9477, 9419
GDS310   2541   9494, 9479, 9480, 9484
GDS320   2544   9500, 9367
GDS350   2589   9541, 9511
GDS360   258F   9544, 9510
GEN      0020    758
GETADR   FF73    612,  616, 9277, 9323, 9349, 9356, 9451, 9466, 9489, 9505,
                9519, 9533, 9628
GETBUF   04CB   1825, 1099, 1118
GETDC1   0530   1904, 1288, 8757
GETDCM   04FC   1872,  995, 1293
GETDSP   242C   9298,  991, 5407, 9340, 9430
GO       00B4   1010,  909
GO010    00D8   1028, 1024
GO1      00DA   1033, 3938
GTB005   04DA   1831, 1843
GTB010   04DD   1838, 1829
GTB100   04F2   1849, 1828, 1841
GTBLK    054D   1943, 2890, 2917
```

```
SYMBOL   VALUE   REFERENCED ON
======================================================================
GTBLKF    054B    1940, 2032, 3022, 3196
GTF010    1553    5885, 5876
GTFCTK    1542    5872, 1302, 5609, 5643
GTMOD1    1053    4718, 9446
GTMODE    1059    4722, 1005, 5418, 5443, 5600, 6320
GTNWLN    0597    2029, 2093, 2500
H         0048     321
HANGU0    1254    5195,  998, 1101, 1845, 1912, 4437, 5262, 6048, 6563
HDC100    110E    4918, 4931
HDC200    112A    4936, 4925
HDC210    1136    4942, 4940
HNDSHK    0040      43, 5355, 5356, 6276
HNG010    125C    5200, 5202
HOL       0010     694
HOLCNT    FF51     776,  777
HRDER1    0010     737
HRDERR    0004     735
HTAB      1F82    8320, 9739, 9816
HTB100    1F9C    8341, 8384
HTB120    1FA1    8344, 8371
HTB130    1FBD    8369, 8346
HTB140    1FC1    8376, 8343
HTB160    1FA5    8350
HTB200    1FCE    8389, 8322
HTBCLR    1560    5902, 9796
HTBLEN    000A     604,  605, 4887
HTBSET    155A    5894, 9795
HTBTBL    FF78     605,  609, 4886, 5773, 6797
HUP050    1D7D    7839, 7798
HUP060    1D92    7854, 7777
HUP100    1DA3    7865, 7782
HUP110    1DA6    7867, 7843
ICH010    1FD7    8401, 8417
ICHOFF    1FDC    8405, 9825
ICHON     1FD5    8399, 9824
IDG055    25BE    9596, 9583
IDG060    25D3    9606, 9601
IDG070    25D7    9609, 9604
IDG090    25E8    9616, 9611
IDG100    25EE    9623, 9603
IGNTRM    0001     656, 2317, 2531, 6401, 6611, 8099, 9597
INERMS    0F3F    4452, 6045
INI010    00FB    1053, 1056
INI020    0103    1062, 1065
INI110    0166    1121, 1117
INI130    0177    1130, 1126
INI210    019A    1169, 1184
INI220    01A1    1174, 1172
INI310    01C8    1206, 1222
INIT      00F4    1048, 1017, 1021, 1027
INITD0    2593    9563,  990
INITD1    2605    9634, 7406, 9149, 9246, 9322
INITDG    259C    9567, 5395, 6447, 9565
INITDS    05CB    2092, 1190, 1236
INPDEV    FF4E     784,  785
```

```
SYMBOL  VALUE  REFERENCED ON
=====================================================================
INSCHR   0002    112, 6972, 8402, 8408, 8521, 9024
INSWRP   0002     81, 7448, 8406, 8414
INTERR   15D7   6039, 6059
INTFLG   FFF6    151,  152, 2621, 2634, 5298, 5301
INTRPT   15EB   6057,  921,  953,  961
INTVEC   9165    136,  137, 1090, 2583, 5152, 6004, 6058
INVRS    0082    411, 5755
IOBASE   0080    392,  396,  404,  416,  425,  432
IOBNGO   1568   5912, 9849
IOBSYC   156E   5918, 2850, 3934, 4114, 5925
IOBUF    FC00    493,  494,  495, 4185, 4214, 4291
IOBUF1   FC00    496, 1061
IOBUF2   FD00    497, 4421, 4429
IOBUFH   00FC    494,  495
IOBUFL   0000    495
IOCCNT   FFD5    805
IOCDEV   FFDB    800
IOCDPT   FF4C    786,  787
IOCERR   FF4F    781,  784, 4040, 5924
IOCINP   FFD9    802
IOCKEY   2802    872,  873, 5739
IOCMND   FFD7    804
IOCNTL   281A    883,  884, 5930
IOCOUT   FFDA    801
IOCRCL   8700    405, 1544, 6008, 8796, 9378, 9487
IOCRRW   8720    406, 1514, 2854, 2989, 3087, 3319, 3794, 4184, 4509, 4516,
                6173, 7717
IOCSGN   FFDD    166,  167, 1737, 4641, 5216, 5248
IOCTCO   8B00    417, 2614, 6047
IOCTDI   8B20    420
IOCTDO   8B20    419
IOCTGO   1580   5929, 9876
IOCTMN   1586   5935, 1325, 6547, 9120
IOCTSI   8B00    418
IOCTO    8B00    416,  417,  418,  419,  420
IOCTYP   FFD8    803
IODATA   FFDE    165,  166,  672, 4644, 4662, 5227, 5234, 5312, 6040, 6424,
                8637, 8645
IODISP   8700    404,  405,  406
IODNGO   2820    885,  886, 1700
IOERRB   0008    380, 4043
IOFLG2   FF64    712,  720, 3328, 4719, 9425, 9530, 9574
IOFLGS   FF65    700,  712, 1312
IOI010   15D1   6025, 6028
IOI020   15D6   6029, 6023
IOINTR   15AD   6003,  944
IOKB     8300    396,  397
IOKBCO   8380    397, 1516, 1618, 4521
IOKEYS   158C   5944, 5328, 5712, 5713, 5714, 5715, 5716, 5717
IOKYTB   14CE   5738, 5945
IOORG    2800    871,  872, 1123
IOPSGN   FFDC    167,  168, 4642, 4646
IOPTR1   8D00    425,  426,  427,  428
IOPTR2   8500    432,  433,  434,  435,  436
IORDGO   2823    886,  887, 1681
```

```
SYMBOL  VALUE  REFERENCED ON
=========================================================================
IORMG1   15A3   5985, 1124, 4140, 5965
IORMGO   1593   5962, 1001, 1040, 1115, 4101, 5914, 5920, 5931, 5937
IOSTA0   FF48    790,  791
IOSTA1   FF49    789,  790
IOSTA2   FF4A    788,  789
IOSTA3   FF4B    787,  788
IOSTGO   281D    884,  885, 1696
IWRPON   1FE6   8413, 9821
JMP      00C3    369, 1015, 1109, 1269
KBDCSW   FFFC    145,  146, 2138
KBDLOK   0040    648, 6072, 6078, 6100
KBEN     15F4   6070, 4782
KBEN1    15FA   6075, 9846
KBFCTK   FF71    617,  619
KBJMP2   FFFA    147,  148, 4107, 5335, 6450, 9462
KBJMP3   FFF9    148,  149, 9684
KBJMPR   FFFB    146,  147, 2121, 3307, 4009, 4734, 5354, 6275, 6289, 8535,
                8787
KBLOK    160C   6103, 6311
KBLOKO   1607   6099, 9847
L        004C    322, 4542, 8686, 9266, 9334
LADDR    FFD5    671, 6141, 6155, 6159, 6175, 6610, 6688, 6691, 6741, 6751,
                7989, 8039
LCHAR    FF69    665,  666, 1634, 5681
LCHKSM   FFD7    673, 6124, 6144, 6146, 6187, 6604, 6634
LC1050   060A   2148, 2124, 2130, 2134
LDATA    FFDE    672, 6140, 6153, 6185, 6608, 6614, 6754
LDR0     1627   6127, 6147, 6195, 9991
LDR035   1640   6143, 6160
LDR060   165A   6165, 6171, 9994
LDR10    1675   6180, 9992
LDR3     1639   6139, 9990
LDR4     164A   6152, 9993
LDRCHK   0004     91, 6183
LDRMSG   0F4A   4458, 6120
LDRTAB   279C   9958, 6133
LF       000A    290, 1440, 1462, 1898, 5620, 6330
LFPOS    0010     35
LFTBKT   005B    331, 9216
LFTBRC   007B    345, 9212
LFTCTU   0001    764
LFTMGN   FFBF    540,  541, 6368, 6375, 6767, 6816, 6993, 7485, 8793
LI1      27C7   9989, 9960, 9963
LID050   09D4   3128, 3117
LID200   09F0   3168, 3155
LID300   09F6   3177, 3163
LII200   0A61   3296, 3279
LINDEL   09B7   3110, 9820
LINDLO   09DA   3144, 3118, 3718, 3788
LININO   0A27   3223, 3126
LININ1   0A3C   3247, 3219
LININA   0A39   3243, 3722, 3795
LININS   0A00   3192, 7472, 9819
LINWRP   0004     31, 3308, 8536
LLINE    FFA1    556,  557, 1228, 2035, 2053, 2187, 2194, 2508, 4820, 4823
```

```
SYMBOL  VALUE  REFERENCED ON
=====================================================================
LNF100   0A81   3320, 3316
LNFEED   0A6F   3310,  992, 5434, 5454, 8548, 8768, 9517, 9542, 9740
LNKLIM   00D0    366, 3616, 7278, 9359
LNKSAV   FF96    572,  573, 2705, 2732, 2796, 2800, 7357, 7404, 7437, 7521,
                7947, 8036, 8355, 8363
LOADR    1611   6117, 9885
LOADR1   161C   6122, 9886
LOCKKB   0001    209, 6104
LOCLIO   05E4   2120, 1377
LOCLIN   05EF   2128, 1431
LP       0040    728
LPM      0001    722
LSTCOL   FFC8    524,  525, 1531, 3227, 3821, 6740, 7974, 7986
LSTDCD   FFC6    528,  529, 3359, 7952, 7965, 8190, 8978, 9631
LSTFMT   FFC5    529,  530, 2031, 3361, 4853, 6231, 6246, 8076, 8211, 8464,
                8530
LSTFWD   0002    723
LSTLIN   FFC9    521,  524, 2097, 2178, 2450, 2966, 3114, 3216, 3218, 3354,
                3771, 4802, 4922, 6609, 6632, 6653, 6660, 6690, 6693, 6743,
                6752, 7460, 7569, 8038, 8461, 9257
LSTLU1   0A9F   3355, 2528
LSTLU2   0AA0   3357, 3224
LSTLUP   0A9C   3353, 2510, 3815
LSTRED   FF25    830,  831
LSTROW   FFC7    525,  528, 1195, 2447, 2570, 3725, 3765, 3797, 3841, 4919,
                4929, 4939, 6622, 6738, 7848, 7857, 7870, 8035
LWBUF    00B0    579, 1071
LWDSP    00D0    492, 1079
MAXCOL   004F    375, 2100, 2946, 3069, 4964, 6629, 6799, 7263, 7324, 7425,
                7436, 7759, 7920, 7975, 7983, 8139, 8326, 8352, 8514, 8566,
                8580, 8626, 8664, 8666, 8677, 9005, 9325, 9327, 9396, 9483
MAXROW   0017    374, 3315, 3500, 3661, 3769, 4920, 4973, 5044, 5051, 5052,
                5283, 6118, 6704, 7716, 8001, 8585, 8590
MAYEOL   0040    408, 1512, 2987, 3064
MAYEOP   0020    407, 3793
MDFLG1   FFF4    153,  154, 1892, 4078, 4703, 4710, 4829, 6902, 6917, 6968,
                7017, 8520, 8749, 8755, 9023, 9697
MDFLG2   FFF3    154,  155, 1435, 1711, 1873, 2131, 4017, 4723, 5330, 5452,
                5504, 5534, 5617, 6326
MEMLOK   0004    113, 3380, 3394, 3458, 4830, 6919, 9699
MFLGS    FF70    619,  630, 1667, 3325, 5138, 5370, 5541, 6303
MFLGS2   FF6F    630,  641, 1297, 1460, 1679, 1768, 4024, 4770, 4777, 5036,
                5102, 6392, 6978, 9316
MINUS    002D    305
MLK010   0B19   3464, 2209, 3457
MLKFLG   FF6A    664,  665, 3379, 3396, 3455, 9696
MLKOF    10C0   4828, 2292, 3376
MLKOF0   0AB9   3373,  997
MLKOFF   0AC0   3377, 9857
MLKON    0ACB   3385, 9856
MLKROW   FF6B    663,  664, 2233, 3374, 3392, 3415, 3421, 3501, 3505, 3662,
                3723, 3735, 3792, 3796, 4982, 6617, 6637, 6705, 6717, 6726,
                6775, 6920, 7846, 7866, 8010
MLKSC0   0AE4   3414,  996
MLKSC1   0AF6   3428, 5584, 8663
```

```
SYMBOL  VALUE  REFERENCED ON
=======================================================================
MLKSCH   0AEE   3420, 3705, 3780, 7781
ML0005   0AD6   3391, 3388
ML0010   0AD9   3393, 4832
MLOCK    0B07   3454, 1950, 2171, 6468
MLOCK0   0B04   3452, 2236
MLOCK1   0B16   3462, 8861
MLS120   0AF7   3430, 3436
MNMDON   1470   5655, 9861
MOVCHR   0B20   3484, 1201, 3492, 7711
MSGPT1   FFF1    155,  156, 5968, 7697, 7701, 7725
MSGPT2   FFEF    156,  157, 4433, 6041
MSGPT3   FFED    157,  158, 4430
MSGPT4   FFEB    158,  159, 4419, 4422
MSGPT5   FFE9    159,  160
MSGPT6   FFE7    160,  161
MSGPT7   FFE5    161,  162
MSGPT8   FFE3    162,  163
N        004E    323, 4540, 5596, 8684, 9263
NBLKS    FF99    569,  570, 2909, 2927
NCH010   0B99   3629, 3617
NCHAR    FF9B    567,  568, 2882, 2896, 2978, 3004, 9057, 9072
NEWCOL   FFDB    597, 4952, 4965, 5016, 8758
NEWROW   FFDA    598, 4988, 5039, 7896, 7903, 7949, 7995
NEXTPG   0B2D   3499, 9828
NMFCTK   0008    400, 1202, 8619
NMPNDG   0008   1702, 1669
NMROLL   FF83    594,  595, 5062
NODCST   0010     69
NODRVR   0F59   4466, 5967
NOFNCT   1971   6879, 9741, 9742
NORMAL   0080    412, 5755
NOSEND   0004    645, 6484, 6506, 9570
NOSIGN   0080    384, 5220
NOTEST   0004     58, 4108
NOTSMS   0F51   4463, 4109
NROWS    FF9A    568,  569, 2174, 2333, 2336, 2491, 2502, 3466
NULL     0000    289
NUM2K    0800    367, 4125, 4128, 4152
NUMBER   00C6    359, 8234
NUMSWP   000F    542,  543, 8724
NWRWST   0080    639, 4953, 4985, 5037
NXB060   0B67   3576, 3591
NXB100   0B75   3583, 3574
NXB150   0B7E   3588, 3586
NXB200   0B83   3592, 3579, 3582
NXSBLK   0B5E   3570, 4198, 4280, 4304
NXT040   0B35   3504, 3665
NXT100   0B45   3523, 3503
NXT110   0B4A   3529, 3536
NXT120   0B59   3542, 3534
NXTCH0   0B86   3611, 4937, 5875, 7096, 8675
NXTCHR   0B87   3613,  994, 2368, 2389, 4869, 7053, 7100, 7171, 7190, 7408,
                7552, 7638, 8068, 8161, 8291, 8883, 9626
NXTPG1   0B45   3524, 5054
NXTRED   FF27    829,  830
```

```
SYMBOL  VALUE  REFERENCED ON
=========================================================================
NZEXIT   0B01   3438, 2033, 2490, 3418, 8071, 8176, 9615
OCTRDX   0008    131, 6134, 6418
OPSTOR   FFD0    504,  505,  512
OTHER    FF56    749,  751
OUTDEV   FF4D    785,  786, 1106
P        0050    324, 5988
PAGSTR   0008     33, 4735
PARM1    FFDB    168,  169,  597,  800, 6226, 6234, 8627, 8669
PARM2    FFDA    169,  170,  598,  801, 8613, 8682
PARM3    FFD9    170,  171,  599,  802, 8620, 8658, 8668, 8700
PARM4    FFD8    171,  172,  803
PARM5    FFD7    172,  173,  673,  804, 7403, 7435
PARM6    FFD5    173,  174,  671,  805, 7410, 7421
PAROT1   0B9E   3647
PAROT2   0B9D   3645, 4005, 9675
PAROT3   0B9C   3643
PAROT4   0B9B   3641, 4013, 4034, 9688, 9692
PAROUT   0B9F   3649, 4011, 4020, 4048, 4064, 9680, 9686, 9690, 9700
PERIOD   002E    306
PLUS     002B    303
POLL     0040    106, 6006, 6010
PRCCTL   FFF5    152,  153, 1012, 1088, 2584, 5293, 6005
PREND    169E   6220, 9834
PREVPG   0BA9   3660, 9829
PRINTR   0008    767
PRM010   2749   9883, 9870
PRMSEQ   168E   6201, 9767
PRMTAB   2732   9868, 6202
PRNTAL   0010     60
PRO010   16A0   6225, 6216
PRO100   16B9   6239, 6210
PROCSR   0070    391, 1011, 2588, 2590, 5292, 5294, 6007, 6011
PROFLD   FFC2    532,  538, 3365, 4743, 5801, 8101, 8109, 8217, 9607
PROMPT   000D    270, 5142
PRSTRT   1694   6208, 6233, 9836
PRV100   0BB6   3684, 3664
PRV110   0BB9   3686, 3691
PRVPG1   0BB4   3671, 5043
PTB090   0644   2208, 2250
PTB100   0647   2210, 2172, 2274, 3453
PTB200   0651   2221, 2177, 2180
PTB220   0671   2245, 2227, 2240
PTB300   068C   2272, 2204
PTBLK    0613   2169,  981, 1949, 5840
PTDLY    05DC    473
PTR120   0212   1265, 1250, 1257
PTRABT   FE78    857,  860
PTRBBG   FE7D    854,  855
PTRBD2   001F    486, 1260
PTRBLN   0100    499
PTRBPT   FE79    856,  857
PTRCF2   8540    436, 1259
PTRCL1   8D02    428, 1248
PTRDA2   8560    435
PTRDY1   0001    477
```

```
SYMBOL   VALUE  REFERENCED ON
===========================================================================
PTRDY2   0002     482
PTRFLG   FE77     860, 1266
PTRHD2   00E0     485
PTRI10   0200    1254, 1247
PTROL2   0020     484
PTROI1   8D20     426
PTROI2   8540     433, 1262
PTRPO1   0080     478
PTRSB2   0040     483
PTRSPT   FE7B     855,  856
PTRST1   8D00     427, 1245
PTRST2   8520     434, 1255
PTTPLN   2832     894,  895, 2249
PUTBRK   0005     262, 5161
PUTLIN   0691    2291, 3127, 7617
QUOTE    0027     301
R        0052     325, 5109
RADIX    FFD4     174,  175, 1734, 5226
RAMERR   0F3B    4449, 4402
RC4010   06EF    2388, 2399
RCA120   0721    2464, 2469
RCA130   072D    2475, 2483
RCA140   0735    2480, 2458
RCA200   073A    2488
RCA210   0746    2499, 2504
RCA220   0754    2509, 2470, 2479
RCA240   075E    2519, 2451
RCA245   0769    2530, 2513
RCA250   0771    2538, 2523
RCA255   077B    2545, 2533
RCA260   077C    2552, 2543
RCA270   079B    2567, 2565
RCA440   06F7    2396, 2379, 2385
RCA460   0702    2401, 2391
RCADDR   0708    2442, 5022, 8995
RCADR0   070B    2444, 2335, 3335, 3743, 5060, 6229, 8860
RCADR1   06B4    2327, 3112, 3194, 6613
RCADR2   06B8    2330, 2322
RCADR3   06BB    2332, 2364
RCADR4   06CD    2361, 4838, 6995, 7022, 7233, 7539, 9599
RCADRA   06A4    2315, 1004, 3910, 6481
RCADRB   06AC    2319, 7904
RCKYCD   009D    5720, 5327
RCRDGO   2826     887,  888, 1899
RCVMDE   0020      85, 1339
RDABRT   2837     896,  897, 1314
RDWOWT   0001     702
RDY      0040     692
REC      0008     756
RECINI   0010     707
RECKEY   280B     875,  876, 5741
RECORD   0040     117, 1893
RECPGE   0020     708
RECRWD   0008     705
RECSEP   5003     232,  233, 5422
```

```
SYMBOL  VALUE  REFERENCED ON
==========================================================================
REDKEY   2805    873,  874, 5740
RELSNS   0004    634, 5071, 5078, 5103
RELTAK   FF61    741,  743
REMOTE   0008    125, 1874, 2132, 2133, 5503, 5533
REMSET   0010     84, 1339, 1877, 1922, 4751, 5326, 5512, 5546
RESET    0000    389
RET      00C9    370, 1089
REXMIT   0001    381
RGTCTU   0002    765
RHTMGN   FFBE    541,  542,  543, 1423, 2101, 6363, 6803, 6839, 6962, 7000,
                7078, 7266, 7452, 7468, 7760, 8356, 8509, 8726
RIP      0004    696
RLCRSN   11D1   5070, 9844
RLD080   0BEF   3734, 3706
RLD085   0C0A   3752, 3737
RLD090   0C18   3766, 3727
RNGTA    FFD2    175,  176, 1571, 1735, 1754, 3886
ROL080   0C66   3828, 3781
ROL090   0C74   3842, 3799
ROL100   0C51   3802, 3845
ROL200   0C53   3804, 3774
ROLLCT   FF82    595,  605, 3526, 3532, 3685, 3688, 6606, 6641, 6646, 6679,
                6697, 6707, 6713, 8017, 8023
ROLLDN   0BC5   3704, 3687, 3748, 6709, 6779, 9827
ROLLUP   0C27   3779, 2235, 3321, 3531, 4921, 8022, 9826
ROLUP1   0C6E   3839, 7874
ROLUP2   0C54   3806, 4928
ROLUP3   0C56   3812, 7840
ROLUPC   0C57   3814, 7860
ROMERR   0F37   4446, 4160
RPTKEY   0003    211
RSETDC   0002    259, 1037
RSETKB   0007    215, 1035
RSTCTU   2817    879,  883, 1039
RSTDSP   1D0E   7747,  968, 1041, 3908
RSTJMP   0001    390
RSTOFF   0004    399, 1515, 1617
RSTON    0002    398, 4520
RSTTMR   FFD0    179, 1022, 2591, 3937
RTABLE   2664   9722, 1749, 3885
RTB010   2677   9735, 9726
RTB020   2679   9737
RUN      0001    753, 5922, 6170
RXMERR   0F42   4455, 4432
S        0053    326
SAVINP   FF23    834,  835
SAVOUT   FF22    835,  839
SBINRY   0002    633, 4778, 5542
SBL010   16E2   6296, 6277
SBL020   16E3   6298, 6292
SBLXF0   16CA   6267,  975, 3971, 4781, 5082, 9644
SBLXF1   16D5   6288, 5347, 5369, 5603
SBLXFA   16CD   6274,  976, 5359, 5602
SCHRST   0C7C   3853, 9770
SCHST1   0C82   3859, 9904
```

```
SYMBOL  VALUE  REFERENCED ON
===========================================================================
SCNCNT   FF54    772,  773, 1320
SCNVEC   9168    137, 1091, 1324
SCRNRW   FFD9    599, 4955, 4974, 5004, 8599
SCRSEN   1000    625, 4032, 5081, 5085
SDACOM   0001    643, 1335, 1650, 1709, 4435, 4796, 5276, 6343, 6415, 7770
SDC2     0100    621, 5140, 5371, 6291
SDTER1   1220   5129, 5461
SDTERM   121D   5127,  985, 3986, 5644, 5648, 9655
SDTRM1   16F6   6319,  986, 5128, 5459
SDTRM2   16FF   6323
SDTRM3   170A   6329, 5432
SDVDON   8000    628
SDVREC   0001    632, 4778, 5542
SDVST    0800    624
SELECT   0020    116
SELKEY   280E    876,  877, 5742
SENTER   4000    627, 3326, 4032, 5329, 5346, 5465, 6253
SESCTB   2686   9750, 1719
SETCH    0020     71
SETDF0   170F   6342, 1891, 6573, 9089, 9568
SETDFL   1711   6344, 5599, 6101, 6507
SETFRN   000C    220, 7555
SETLCL   0004    261, 5510
SETLFT   1717   6360, 9798
SETMF2   1739   6391, 1714, 4986, 5072, 6943, 9274
SETMON   0008    265, 5656
SETNRM   0009    266, 5686
SETREM   0003    260, 5544
SETRHT   1729   6371, 9799
SETROM   0080    107,  908, 1087, 5291
SETTRG   0001    258, 1657
SETTRM   173F   6400, 6475, 9838
SEVEN    0037    314,  960
SFCTKY   2000    626, 4032, 5601, 5634
SFKCHK   0FB7   4536, 8238
SFKYAT   00C8    361, 5754, 9225
SFKYDS   0CBA   3921, 1648, 9718, 9953
SFKYOF   0C8D   3878, 3925, 4613, 5678, 5698, 7724, 9855
SFKYON   0CA5   3898, 9854
SF0010   0CAE   3906, 3884, 3887
SFTCNT   FF5D    748,  749
SFTDLY   0032    383, 1025, 1029
SFTEND   0010    376, 9241, 9585
SFTERR   0008    736
SFTKYS   FFA6    553,  554, 1192, 5583, 7858, 8662
SFTRST   0CCC   3933, 9851
SHFT1    0CDF   3947, 3964
SHFT2    0CE6   3953, 4334
SHFTIN   0CEB   3960, 9745
SHFTOT   0CD8   3943, 9744
SI       000F    294, 9204
SIX      0036    313,  952
SKPTRM   0008    646, 6486, 9381
SLANT    002F    307
SLKYCD   009E   5721
```

```
SYMBOL   VALUE  REFERENCED ON
=====================================================================
SMALLA   0061     337, 5089, 9271
SMALLD   0064     339, 9160
SMALLF   0066     340, 9248
SMALLI   0069     341
SMALLK   006B     342, 9255
SMALLP   0070     343
SMALLX   0078     344, 1213
SNDATN   000B     268, 6426
SNDCD1   175A    6423, 9979
SNDCD2   12CE    5311, 9982
SNDCDE   174C    6413, 9890
SNDCTB   27B0    9974, 6417
SNDFCI   000C     269, 5314
SO       000E     293, 9200
SPLDIS   0002      29, 8788
SPOWL    FF6C     658,  663, 1281, 1494, 3311, 6895, 7780, 8790, 8951
SPOWOF   00FF     661, 3312, 6896
SPOWON   0020     660, 8791
SSTAT    0200     622, 3970, 3976
SSTAT2   0400     623, 4026, 9643, 9649
STA010   0D51    4044, 4042
STA2G1   2626    9666, 9654
STA2G2   2629    9668, 4386
STA2GO   2612    9648, 1695
STACK    9160     503, 1013, 1287
STAPAR   0D14    3997, 3981, 4384
START    0220    1275, 1043
STAT2    260C    9642, 9786
STATGO   0CF9    3975, 1694
STATUS   0CF3    3969, 9837
STB010   177D    6465, 6462
STB050   1784    6474, 6464
STB060   1799    6483, 6505
STB080   17A2    6493, 6480
STBLMD   0004     212
STC010   0D7A    4087, 4081
STCHR1   0D68    4077, 2055, 3208
STCHST   000D     221, 5400
STCMFL   1400    5555, 2852, 3902, 5198, 6126, 8415
STFOR1   00FE    5733, 1382
STFOR2   00FD    5732, 1384
STOREA   1605    6093, 3770
STPFLG   00C4     357, 4875, 6403, 6478, 7667, 7937, 8074, 8201, 8886, 8891,
                 9416, 9625
STPR     00C0     353, 2030, 3360, 5752, 6209, 6232, 7068, 7290, 7665, 7914,
                 7914, 7956, 8097, 8097, 8216, 8939, 8965, 9437
STPRPT   0009     217, 9125
STPXFR   FFFF     654, 5397, 9612
STR010   11BF    5053, 5063
STRTAK   405F     745
STRTB1   1769    6449, 6446
STRTBL   1763    6445,  977
STRTST   0005     213, 4116
STRXMO   1699    6214, 9863
STTERM   1771    6460, 5338
```

```
SYMBOL  VALUE  REFERENCED ON
=====================================================================
SWAP     2169   8718, 1232, 3907, 8657, 8748
SWAP0    2163   8713, 8691, 8756
SWAP1    216F   8723, 2213
SWCHAR   000B    219, 3949
SWP010   2177   8730, 8740
SWPCTU   FF24    831,  834
SWPSTR   FFAF    543,  544, 8725
T        0054    327, 4544, 8688
TAK      0008    695
TCHAR    FF68    666,  667, 4328, 4336, 4342
TEMP     FF9D    565,  566, 7370, 7379, 7953, 7964, 8062
TEMP1    FF9E    564,  565
TEST     0D7D   4095, 9862
TESTOK   0002     90, 4381
THREE    0033    310,  928
TKI      0080    691
TLINO    FFA3    555,  556, 3719, 3789, 4479, 4990, 5040, 5107, 7778, 9441,
                9513
TMI010   07C2   2598, 2594
TMI020   07CA   2604, 2602
TMI100   07DB   2620, 2607
TMI110   07F7   2636, 2624
TMIACK   0000    100
TMIEN    0002    103,  908, 1087, 2587, 2589
TMINIR   07A4   2582,  929
TMIOFF   0020    105
TMPCOL   FF85    589,  593, 2445, 2511, 2520, 2563, 6631, 6694
TMRINT   0003     95, 2622, 5302
TMRON    0001    102,  908, 1087
TOP100   0F85   4484, 4482
TOPLIN   FFCB    518,  521, 2099, 2222, 3417, 3427, 3711, 3753, 3791, 3803,
                3829, 4489, 6681, 7748, 7793, 7820
TOPUP1   0F86   4486, 3231
TOPUPD   0F79   4475, 3764, 3840
TPSTAL   FF50    777,  781, 2599
TRIGGR   5002    231,  232, 1652
TRMFCI   FF6D    652,  658, 2318, 2532, 2556, 5398, 5463, 6402, 6612, 8100,
                8106, 8249, 9598, 9613
TRMRDY   0F6A   4469, 1267
TRMTST   0D8C   4106,  988
TRMTYP   FFFD    144,  145, 1128, 9679
TST010   0DAB   4127, 4137, 4139, 4144, 4156
TST020   0DCE   4150, 4141
TST030   0DD9   4159, 4146
TST050   0DE8   4182, 4135
TST060   0DF9   4197, 4201
TST090   0E06   4209, 4282
TST100   0E09   4215, 4224
TST115   0E13   4227, 4230
TST120   0E1A   4235, 4245
TST125   0E27   4248, 4251
TST130   0E2C   4256, 4267
TST140   0E42   4279, 4274
TST150   0E55   4303, 4311
TST160   0E68   4314, 4307
```

```
SYMBOL  VALUE  REFERENCED ON
=========================================================================
TST200   0E72   4324, 4352
TST220   0E79   4329, 4346
TST240   0E84   4335, 4348
TST420   0EAF   4359, 4370
TST440   0EC4   4372, 4367
TST500   0EED   4397, 4312, 4317
TST510   0EEF   4400, 4239, 4261
TST600   0EF3   4414, 4169
TST610   0F0E   4428, 4420
TSTCTU   2811    877,  878, 4100
TWO      0032    309,  920
TYPSET   0FAB   4527, 9774
U        0055    328
UNITO    FF63    720,  731
UNLKKB   0002    210, 6076
USL      0010    757
USREAD   0002    703, 1313
VERIFY   0080    710
VERSN    0050      5,  905, 4607, 6583, 8434
VERSN1   0051      6, 2649
VRTBAR   007C    346, 9651
WBSR     0020    126
WRPDEL   0020    637, 6942, 6980
WRPFLG   0040    638, 1465, 1477, 8540
WRTERR   0020    738
WTL010   0236   1292, 1322, 1327, 1342
WTL020   025F   1319, 1299, 1311
WTL200   0274   1332, 1303, 1305, 1441
WTL205   028C   1344, 1340
WTL210   02A6   1362, 1350
WTL250   02B9   1374, 1364, 1366
WTL260   02BB   1376, 1391, 1396, 1401, 1406
WTL270   02DC   1393, 1381
WTL280   02E3   1398, 1383
WTL290   02E9   1403, 1385
WTL300   02F0   1412, 1348
WTL310   0313   1430, 1414, 1418, 1421
WTLOOP   0230   1286, 1358, 1369, 1388, 1434, 1437
XBF2DS   0080    649, 4112, 5276, 8539
XDS2BF   0020    715, 3329, 4720, 9426, 9531
XFRLIM   FF47    791,  792
XMD000   0FC8   4560, 4577
XMD010   0FCB   4563, 4580, 4586
XMD020   0FE7   4584, 4575
XMD030   0FED   4590, 4572
XMOHME   17F4   6572, 5379, 6452, 9815
XMONLY   00C2    355, 2397, 6215, 8134, 8205, 8493, 8873, 9211
XMS2DS   0FCD   4567, 1219, 7733
XPD001   17C9   6529, 6553
XPD005   17D4   6535, 6528
XPD010   17D6   6543, 6534
XPD050   17EA   6560, 6533
XPUTDC   17C1   6524,  987, 2136, 3980, 5090, 5094, 5098, 5121, 5402, 5409,
                5423, 5431, 5442, 5649, 6322, 6325, 6331, 6521, 9653
XTRASP   FE80    845,  847
```

```
SYMBOL  VALUE  REFERENCED ON
=======================================================================
Y        0059    329, 5105
Z        005A    330
ZALPCK   4823    199,  200, 8231
ZBELL    4814    194,  195, 1341, 1426, 3463, 4322, 5191, 6230, 6564, 8495,
                9015, 9034, 9118, 9135, 9736
ZBRK1    0800   2648, 4605, 2650
ZBRK2    1000   4606, 6581, 4608
ZBRK2C   1002   4609, 4604
ZBRK3    1800   6582, 8432, 6584
ZBRK4    2000   8433, 8435
ZCLMD1   4811    193,  194, 3381, 5690, 5802, 8409
ZCLXMT   481A    196,  197
ZCTLAL   6014    282,  283
ZDCBAS   5000    230,  231,  241
ZDCCTL   5011    244,  245, 5188, 5687
ZDCINT   5026    251, 5154
ZDCMON   500E    243,  244, 2631
ZDCTST   5014    245,  246, 5261
ZDSPMS   0040    967
ZERO     0030    308, 1207, 2678, 2746, 2766, 2787, 3651, 4531, 5223, 9230,
                9253, 9262
ZGETAL   600E    280,  281
ZGETDC   5017    246,  247, 1883, 6167
ZGETKY   4805    189,  190, 1304, 9121
ZGTBIN   501D    248,  249
ZIN2AL   6005    277,  278, 1119
ZIN2DC   500B    242,  243, 1100
ZINIAL   6002    276,  277, 1114
ZINIDC   5008    241,  242, 1098
ZINIKB   4802    188,  189, 1097
ZINTAL   6008    278,  279, 6016
ZKBBAS   4800    187,  188
ZKBCTL   4808    190,  191, 1036, 2597, 3950, 4099, 4117, 4393, 5323, 5401,
                5850, 5856, 6077, 6105, 6549, 7556, 9126
ZKBMON   480B    191,  192, 2629
ZMONAL   600B    279,  280, 2628
ZMSGAL   601A    284
ZNDBIN   5023    250,  251
ZNUMCK   4826    200,  204, 8233
ZPUTAL   6011    261,  282
ZPUTDC   501A    247,  248, 6532
ZRETRN   0706   2407, 3366, 7905, 8208, 8236
ZSTAAL   6017    283,  284
ZSTBIN   5020    249,  250
ZSTJPR   481D    197,  198, 9879
ZSTLKY   4820    198,  199, 9873
ZSTMD1   480E    192,  193, 3398, 3461, 5668, 7762, 8403
ZSTXMT   4817    195,  196
  1348 SYMBOLS,  4588 REFERENCES,    47 WORK TRACKS
```