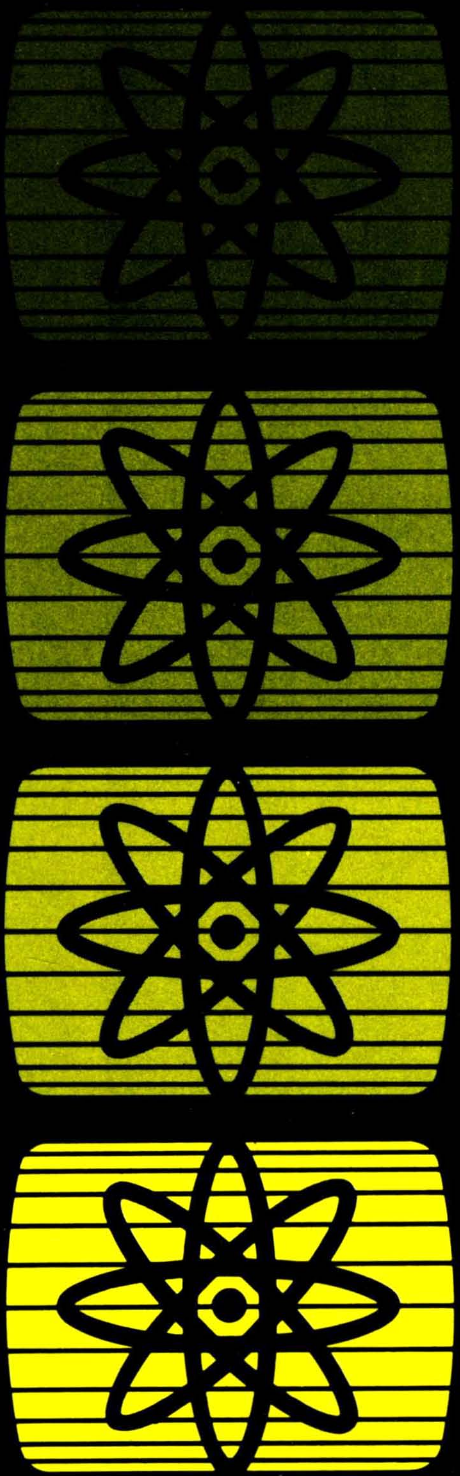


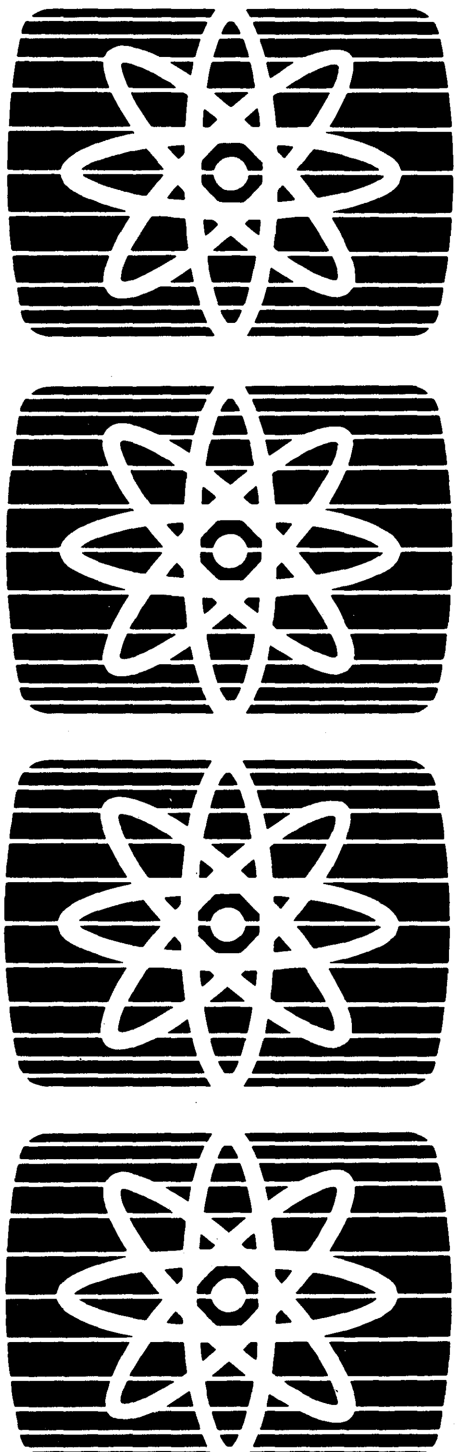
7171

ASCII Device Attachment Control Unit

Reference Manual and Programming Guide

GA24-4020-0

IBM



7171

ASCII Device Attachment Control Unit

Reference Manual and Programming Guide

GA24-4020-0

IBM

Federal Communications Commission (FCC) Statement

Warning: This equipment generates, uses, and can radiate radio frequency energy and if not installed and used in accordance with the instruction manual, may cause interference to radio communications. It has been tested and found to comply with the limits for a Class A computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a commercial environment. Operation of this equipment in a residential area is likely to cause interference in which case the user at his own expense will be required to take whatever measures may be required to correct the interference.

First Edition (October 1984)

This edition, GA24-4020-0, applies to the initial release of the IBM 7171 ASCII Device Attachment Control Unit Reference Manual and Programming Guide, and to all subsequent releases of this product until otherwise indicated in new editions or Technical Newsletters.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this publication is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the address given below; requests for IBM publications should be made to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Engineering/Scientific Systems, Department X00, P.O. Box 6, Endicott, NY, U.S.A. 13760. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Preface

This publication describes the functional capabilities and the operation of the "IBM 7171 ASCII Device Attachment Control Unit," referred to in this publication as the "IBM 7171." The publication is intended for system programmers and operation personnel, but also contains useful information for terminal users. The following is an overview of the contents of the publication:

Chapter 1. Introduction to the IBM 7171 - Gives general overview and uses of the system.

Chapter 2. System Description and Operation - Defines the main components and their functions. Describes the setup requirements and system operation.

Chapter 3. Using ASCII Terminals on the IBM 7171 - Describes the system functions that are available for various terminal types.

Chapter 4. Customizing IBM 7171 Tables - Describes how to modify supplied Terminal Definition Tables and generate new ones for new terminal types. Explains how to modify the predefined configuration parameters for each port address.

Chapter 5. IBM 7171 I/O Interface to Terminals - Describes the Terminal Controller I/O interface protocol to devices attached via RS-232-C lines.

Chapter 6. IBM 7171 I/O Interface to the Host System - Describes the Channel Controller interface protocol to the host computer channel.

Chapter 7. Diagnostic Program - Describes the Diagnostic Program operation.

Chapter 8. Problem Determination - Explains how to handle problems that may occur during operation.

Chapter 9. Special Maintenance Facility and System Messages - Explains how to use the special Maintenance Facility. Defines all of the IBM 7171 system messages.

Appendix A. ASCII and EBCDIC Data Conversion Tables - Defines the data conversion tables.

Appendix B. IBM 7171 Supplied Terminal Definition Tables - Defines the Terminal Definition Tables supplied with the system and describes the functions of specific terminals.

Appendix C. IBM 7171 Support Utility for Modifying Terminal Tables - Describes how to use a supplied utility that runs on the IBM Personal Computer that may be used to modify Terminal Definition Tables.

Appendix D. Hints on Using ASCII Printers and Plotters - Gives information about user written line drivers and setting up ASCII printers and plotters.

Appendix E. Interface for User Supplied Table Modification Program - Describes how to add a user supplied program for customizing the IBM 7171 Terminal Definition Tables.

Appendix F. ROM Data Base Organization - Describes the organization of Read-Only Memory and the contents of the various tables.

The reader is expected to be familiar with either the IBM host VM or MVS concepts and terminology. Helpful references are listed in "Related Publications."

Related Publications

IBM 7171 Publications

IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide, GA24-4019.

VM/SP (Release 3) Publications

VM/SP Planning Guide and Reference, SC19-6201.

VM/SP Installation Guide, SC24-5237.

VM/SP CP Command Reference for General Users, SC19-6211.

VM/SP Remote Spooling Communications Subsystem Networking Program Reference and Operations Manual, SH24-5005.

VM/SP System Product Editor Command and Macro Reference, SC24-5221.

VM/SP CMS Command and Macro Reference, SC19-6209.

VM/SP Operator's Guide, SC19-6202.

VM/SP Terminal User's Guide, SC19-6206.

VM/SP System Programmer's Guide, SC19-6203.

MVS (Release 3.8) Publications

OS/VS2 System Programming Library System Generation Reference, GC26-3792.

OS/VS2 MVS Utilities, GC26-3902.

OS/VS2 TSO Terminal User's Guide, GC28-0645.

OS/VS2 TSO Command Language Reference, GC28-0646.

OS/VS2 Supervisor Services and Macro Instructions, GC28-0683.

OS/VS2 JCL, GC28-0692.

Other Publications

IBM 3101 Display Terminal Description, GA18-2033.

An Introduction to the IBM 3270 Information Display System, GA27-2739.

IBM 3270 Information Display System Data Stream Programmer's Reference, GA23-0059.

IBM 3270 Information Display System Reference Summary, GX20-1878.

IBM 3270 Information Display System 3274 Control Unit Description and Programmer's Guide, GA23-0061.

IBM 3270 Information Display System Character Set Reference, GA27-2837.

IBM System/370 Principles of Operation, GA22-7000.

IBM Series/1 Yale ASCII Terminal Communication System II Program Description/Operations Manual, SB30-1911.

Contents

Figures

Chapter 1. Introduction to the IBM 7171

1.1 Overview

The IBM 7171 ASCII Device Attachment Control Unit (IBM 7171) provides the ability to attach a variety of full duplex ASCII devices directly to IBM 43xx or 308x host processors via a block multiplexer channel. Attachment of up to 64 ASCII devices is supported via the RS-232-C electrical interfaces at speeds of 50 to 19,200 bits per second. This baud rate is limited only by the facilities of the communication link and the ASCII device. Automatic baud rate recognition (autobaud) is available on a per line basis for 300, 600, 1200, 1800, 2400, 3600, 4800, 9600 and 19200 baud.

These devices may be connected directly to the IBM 7171 without modems, or via leased or switched lines using line drivers, modems or acoustic couplers.

The IBM 7171 also provides ASCII to IBM 3270 protocol conversion. The IBM 7171 appears to the host processor as one or two IBM 3274 model 1D control units. The attached ASCII display terminals and printers appear to the host system as IBM 3278 or 3277 terminals and IBM 328x printers.

IBM 3270 emulation allows the IBM 7171 attached devices to communicate with IBM interactive packages while utilizing existing 3270 programs with no host modification required.

IBM 3270 emulation extends the capabilities of the ASCII device by providing 3270 type functions. These functions include: program function keys, program attention keys, light pen simulation, numeric-only input fields, highlighting, protected field skip, unformatted screens and editing functions such as, character insert and delete, forward and backward field tabs, erase to end of field, clear screen and cursor movements.

In addition to 3270 emulation, extended functions have also been included. These functions include: type-ahead capability, enhanced null/blank handling, XON/XOFF pacing, dynamically redefinable home position, special indentation features, and forward and backward column tabbing.

The IBM 7171 is designed for easy customer installation, setup and maintenance. It can be integrated into an established communication network or it can be the keystone for a newly designed network.

1.2 ASCII Device Support

The IBM 7171 allows attachment of a wide variety of ASCII devices. The minimum criteria for device attachment are listed in section "2.6.2 ASCII Display Terminal Requirements" on page 2-11.

Device characteristics are contained in Terminal Definition Tables. Default tables are shipped with the IBM 7171 to support several commonly used ASCII device types, including the IBM 3101.

The customer may use existing tables to attach these common device types. He may also modify existing tables or create new tables to support attachment of other devices. This modification may be accomplished using the special Maintenance Facility of the IBM 7171 or using a support utility. The supplied support utility requires an IBM Personal Computer with the Personal Computer Disk Operating System Version 2.00 (or equivalent) and an Asynchronous Communication Adapter.

Color terminals can be defined to emulate IBM 3279 "basic color." Basic color means that four different colors are assigned to protected/unprotected and highlighted/normal fields. Selection of color is table defined.

A special 3270 order sequence provides for transparent output (ASCII data from the host to the device) to hardcopy ASCII devices such as printers and plotters.

Chapter 2. System Description and Operation

This chapter gives an overview of the functions of the IBM 7171. It describes the setup requirements and system operation. This chapter is organized as follows:

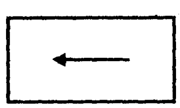
- The IBM 7171 components and functions
- The ASCII devices supported, including display terminals, typewriter terminals, printers, and plotters
- Differences between ASCII and IBM 3270 devices
- Customizing the IBM 7171
- Operating environment of the IBM 7171
- Setup requirements for the IBM 7171
- System start-up and operation.

Figure 2-1 shows a typical system configuration for the IBM 7171.

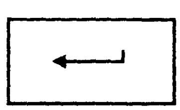
2.1 Terminology

To simplify references to functions that are generated by various keys, or key-sequences, on different ASCII terminals, this publication generally refers to the **name of the function** rather than the key or key sequence used to generate it. For example, "pressing ENTER," or "pressing the ENTER key," should be understood to mean pressing the key, or key sequence, that would generate the Enter function on the specific terminal.

On most ASCII terminals the RETURN key is used to generate the Enter function, but there are several other possibilities. For example, on the IBM 3101 the Enter function is generated by the backspace key:



On the IBM Personal Computer when running the IBM 3101 Emulation Program, the following key is used:



And on the Televideo Model 950, the user can generate the Enter function in three ways: by pressing RETURN, ENTER or the sequence CTRL 'm'. The relationships between the keys on the various ASCII terminals and the functions that they generate are explained in the tables contained in "Appendix B. IBM 7171 Supplied Terminal Definition Tables."

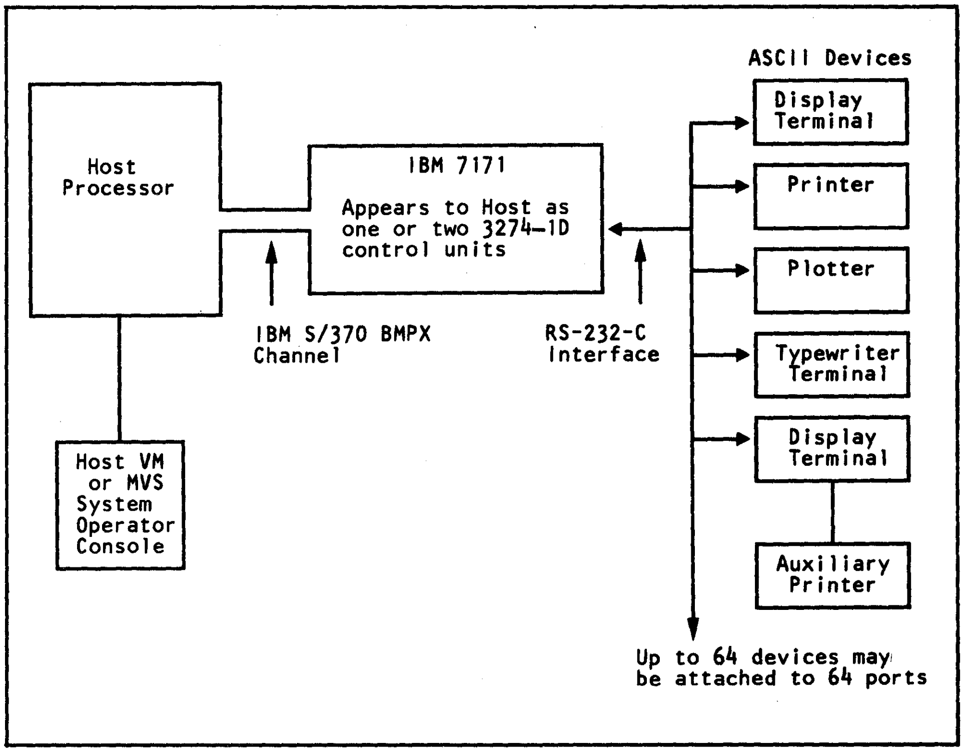


Figure 2-1. System Configuration for IBM 7171

2.2 The IBM 7171 ASCII Device Attachment Control Unit

The IBM 7171 ASCII Device Attachment Control Unit supports a variety of ASCII terminals, printers and plotters. It provides a full duplex asynchronous ASCII terminal interface, with electrical connection via EIA RS-232-C. Flexible communications attachment options allow devices to be connected directly to the IBM 7171 without modems, or via leased and switched lines using line drivers, modems or acoustic couplers. Data rates may be from 50 to 19,200 bits per second, depending on the capabilities of the communication link and the terminal. Automatic baud recognition (autobaud) is selectable on a per-line basis for 300, 600, 1200, 1800, 2400, 3600, 4800, 9600 and 19200 baud upon receipt of a single carriage return (CR) character.

The host processor may be an IBM 43xx or 308x running a VM/SP or MVS operating system. The IBM 7171 attaches to the host via a block multiplexer channel and communicates with the host processor channel program.

The IBM 7171 provides support for ASCII terminals, such as the IBM 3101, for use with interactive host applications running under VM/SP or MVS. It emulates a locally attached 3274-1D control unit with IBM 3270 terminals attached. It allows users of ASCII terminals to communicate with standard IBM host interactive packages and editors operating under VM/SP or MVS. This emulation of 3270 terminals gives ASCII terminal users access to a wide range of host application programs with no host modifications required.

In addition to ASCII terminal support, a transparent interface is provided that passes output from the host to ASCII printers and plotters attached to the IBM 7171.

2.2.1 Hardware Components

The main functional components of the IBM 7171 are the CPU board, the channel controller board, the channel adapter board and up to eight terminal controller boards. Figure 2-2 shows the interfaces between the hardware components. The channel adapter board interfaces with the host processor channel. Each terminal controller controls up to eight ASCII terminals.

When the IBM 7171 power is turned on, microcoded diagnostics verify that the hardware system is operational. A description of the Diagnostic Program may be found in "Chapter 7. Diagnostic Program."

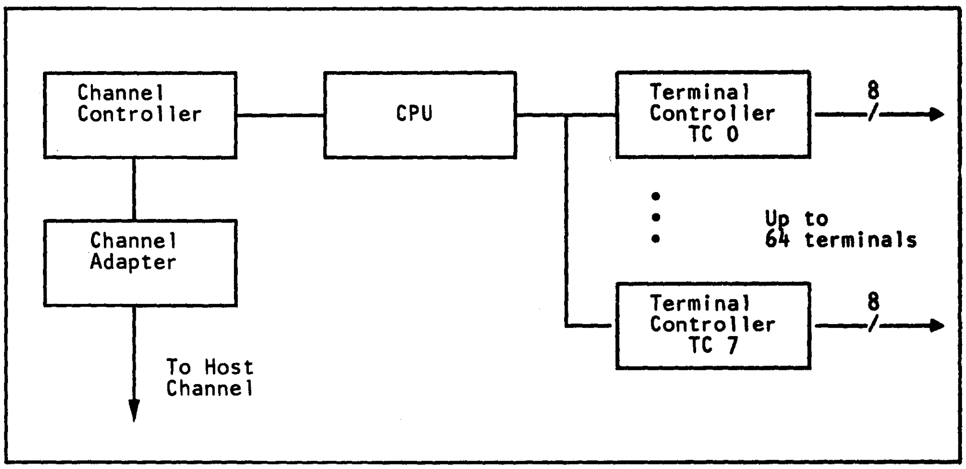


Figure 2-2. Functional Hardware Components within the IBM 7171

2.2.2 Terminal Management

Terminal management within the IBM 7171 for the emulated 3270 terminals is organized around a full screen image buffer, a ring buffer for keyboard input, a buffer for output to the terminal, and Terminal Definition Tables (TDT's) maintained in the IBM 7171 main memory for each ASCII terminal managed by the IBM 7171 Control Program.

The keyboard input routine interprets characters received from each ASCII terminal according to its associated Terminal Definition Table, and separates them into text, local editing function and "attention" functions.

Text characters received from the terminal are entered directly into the screen buffer unless they would alter a protected field. In that case, the terminal's audible alarm sounds and the character is ignored. Local editing functions alter the screen buffer and/or cursor position buffer as appropriate. "Attention" generating functions are passed to the host and do not alter the screen buffer.

The output process is invoked whenever changes are made to a terminal's screen image in IBM 7171 main memory by keyboard input or by output from a user's host application. The terminal screen is updated to match the screen image in IBM 7171 main memory.

2.2.3 Memory Organization of the CPU Board

The main memory of the CPU board contains the Control Program and the supplied Terminal Definition Tables in Read-Only Memory (ROM). A part of memory in Non-Volatile RAM, referred to in this document as NV-RAM, is reserved for the user to add his own Terminal Definition Tables. The integrity of the contents of Non-Volatile RAM is maintained when the IBM 7171 power is off. The memory of the CPU board, which consists of ROM, NV-RAM, and RAM, is organized as shown below.

ROM

- IBM 7171 Control Program
- Diagnostic Program
- Supplied Terminal Definition Tables for several ASCII device types
- EBCDIC/ASCII translation tables.

Non-Volatile RAM (NV-RAM)

- Pointers to Terminal Definition Tables (in ROM and NV-RAM)
- Area reserved for user generated Terminal Definition Tables

RAM

- Screen images for up to 64 active terminals
- Work area for IBM 7171 Control Program
- Host and terminal input/output buffers.

2.2.4 Asynchronous Operation

The terminal input, host input/output, and terminal output processes are coordinated, but operate asynchronously. Also, the user or the host application can generate new screen images more quickly than the terminal is able to display them (for example, by rapid, consecutive page forward requests). The changes occur almost immediately in the screen buffer, but the output process is paced by the terminal line speed. Thus the user may see only the last version of the screen buffer and may not have to wait for all intermediate screens to be displayed.

2.2.5 Error Recovery

Recovery sequences are automatically initiated whenever a transmission or data error is detected. If a transmission error is detected on input data (incorrect parity, framing error or overrun), or when the input ring buffer is full, the terminal is marked in an input-error state. Once marked, every character received is ignored and the ASCII BEL character is sent to the terminal causing a "beep" that informs the terminal user of the error state. The terminal user can exit from this error situation by typing in an appropriate reset key sequence. This error recovery sequence occurs transparently to the host application program.

Should connection with the terminal be lost due to modem or phone line errors, the line is disabled and re-enabled to permit the user to dial back in and logon again. Error situations that a terminal user may encounter at his terminal are described in more detail in section "3.2 Error Situations" on page 3-11.

2.3 ASCII Devices Supported

ASCII devices supported by the IBM 7171 include:

ASCII terminals

- display terminals
- typewriter terminals.

ASCII Output-only devices

- printers and plotters running in transparent mode
- printers emulating IBM 328x printers
- hardcopy devices connected to the auxiliary port of a display terminal.

Display terminals with graphic capabilities can be supported by host applications.

For each IBM 7171 communication line, characteristic data including the line speed (or autobaud detection), parity selection, number of stop bits, type of connection and terminal type must be defined. For a complete definition and the default settings as delivered, refer to section "4.4.2 Ports Area Layout" on page 4-24. The procedure for changing these predefined parameters for each communication line is explained in "Chapter 4. Customizing IBM 7171 Tables."

2.3.1 ASCII Terminals

ASCII Display Terminals

Both IBM and non-IBM ASCII display terminals can be attached. All terminal dependent information is contained in Terminal Definition Tables, which can be customized by the user. Tables for the following terminals, or their equivalents, are provided:

- IBM 3101 (Models 1 or 2 in character mode)
- IBM Personal Computer [1]
- Datamedia 1520/1521/3045 [2]
- DEC VT100 [3]
- LSI ADM 3A/31 [4]
- Televideo 912/920/950 [5].

Hardcopy output of the current screen contents of terminals with local print capability can be obtained on printers attached to an auxiliary device port.

Basically, the attached ASCII display terminals appear to the host system as upper/lower case IBM 3278 or 3277 terminals attached to one or two IBM 3274-1D control units. ASCII APL terminals are emulated as either a 3277 APL display station with the Data Analysis feature or a 3278 APL display station with the APL/Text feature. Refer to section "2.6.2 ASCII Display Terminal Requirements" on page 2-11 for the minimum requirements for attached ASCII terminals.

3270 Emulation: The IBM 7171 provides support for ASCII display terminals when used with interactive host applications running under VM/SP or MVS. As long as existing IBM access methods are used within VM/SP or MVS, the user need not be concerned with the details of the program interfaces to the IBM 7171. Figure 2-3 lists the 3270 features that the IBM 7171 emulates.

Many host applications, such as VM/SP CMS, have formatted screens with specific data entry areas. The screen image that the user sees on a 3270 is almost exactly what he would see on, for example, an IBM 3101 connected via the IBM 7171. The operations that can be performed on the 3270 can also be performed on an ASCII terminal.

For the terminal user, the difference in using an emulated 3270 is minor, and will be noted where it is relevant.

Extended Functions: The additional functions provided in the IBM 7171 make work easier for the ASCII terminal user and are described in detail in section "3.1 IBM 3270 Emulated and Extended Functions" on page 3-1 and in section "6.4.3 Special Order Strings" on page 6-35.

Because of these extended functions, it is sometimes necessary to make explicit references to the differences between the IBM 7171 support and the real 3270. If the user is not already familiar with a 3270 terminal, he may refer to the IBM publications *VM/SP Terminal User's Guide,* or *OS/VS2 TSO Terminal User's Guide,* which are listed under "Related Publications" on page iv.

---

[1] The IBM Personal Computer when running the IBM 3101 Emulation Program uses the standard IBM3101 Terminal Definition Table.

[2] A product of Datamedia Corporation.

[3] A product of Digital Equipment Corporation.

[4] A product of Lear Siegler, Incorporated.

[5] A product of TeleVideo Systems, Incorporated.

KEYBOARD	ENTER ERASE EOF ERASE INPUT INS DEL RESET DUP FM PA 1-3 CURSR SEL PFK 1-36 TEST REQ CLEAR
COMMANDS	WRITE ERASE WRITE ERASE/WRITE ALTERNATE READ BUFFER READ MODIFIED ERASE ALL UNPROTECTED SELECT SENSE NOP
WRITE CONTROL CHARACTER (WCC)	Define Printer Format Start Printer Sound Alarm Keyboard Restore Reset MDT
ORDERS	SF SBA IC PT RA EUA GE
ATTRIBUTES	PROTECTED NUMERIC INTENSITY (terminal dependent) LIGHT PEN DETECTABLE (CUR SEL key) NON-DISPLAY MDT

Figure 2-3. 3270 Features Emulated by the IBM 7171

ASCII Typewriter Terminals

The TYPETERM device type provides limited support of ASCII typewriter terminals. Application programs that write one line at a time to the screen from top to bottom before pausing for input will approximate "roll screen" mode on a typewriter terminal. However, if more than a screen of data is sent, the standard "MORE ..." message will not appear. In this case the typewriter will wait for the user to press an attention key (Enter, PA1, PA2) before typing the next screen, or until the host eventually advances to the next screen. When the terminal type is TYPETERM or HARDCOPY the last line that would appear on a display screen is not printed.

Note: Application programs using the terminal type TYPETERM with the full formatted screen and writing to random locations on the screen, will appear to the terminal user very awkward and difficult to follow.

Terminal Definition Tables

A wide range of ASCII terminals can be handled with the mapping of terminal keys (for example, editing functions and PF keys) specified in Terminal Definition Tables.

These tables allow any explicit sequence of one or more keystrokes to be assigned to each of the editing, "attention" and extended functions. If, for example, the terminal has transmitting cursor movement keys, the codes they generate can be assigned to the cursor movement functions of the system. If the terminal has no PF keys, the "Escape" key sequence and a number key may be assigned to generate the functions. Variations in key placement and hardware capabilities can be used or ignored, depending on installation options. Two similar terminals can be made to function differently, and dissimilar terminals can be made to appear similar, depending on the tables that are used.

All such terminal dependent information is contained in user definable tables. The system as delivered contains Terminal Definition Tables for all ASCII terminals listed in "ASCII Display Terminals" on page 2-5. If, however, the user has other brands of terminals, or he wants to re-define the mapping of terminal keys, he can define new terminal tables as described in "Chapter 4. Customizing IBM 7171 Tables."

During terminal initialization, if the system prompt to enter the terminal type appears, press ENTER and a list of the supported terminal types is displayed. This display of supported terminal types also appears if an invalid terminal type is entered.

2.3.2 ASCII Output-Only Devices

Output-only devices such as printers and plotters may be used in various ways. Host applications can mix normal 3270 order strings with transparent mode output to the same display terminal for graphic output or attached (auxiliary) devices.

Output-Only Devices and User-Written Output Programs

The user can write his own host application program to output data to an ASCII output device using hardcopy transparent mode. The host application can also mix normal 3270 order strings and transparent mode output to the same terminal to drive CRT terminals with graphic capabilities.

Transparent Mode Interface with Host Channel

A special 3270 order sequence is defined to support transparent output (ASCII data from the host to the device) to hardcopy ASCII devices such as printers and plotters. This is described in section "6.4.3 Special Order Strings" on page 6-35.

Printers and Plotters Driven by RSCS Line Drivers

For these printers and plotters, the device type "HARDCOPY" must be set up in the IBM 7171 memory table for this device. Refer to "Chapter 4. Customizing IBM 7171 Tables" for setting up device types.

Line drivers can be used within RSCS under VM/SP to attach ASCII printers and plotters to the IBM 7171. The line drivers communicate with the transparent mode interface of the IBM 7171. Installation of line drivers is optional.

Each printer or plotter served by an RSCS line driver is driven through a separate port address on the IBM 7171-host channel. The host processor operator controls the RSCS line drivers with normal RSCS commands, such as START, DRAIN, and FLUSH. The user responds using CP/CMS commands such as TAG, SPOOL, and PRINT. For information on line drivers to support ASCII printers or plotters, refer to "Appendix D. Hints on Using ASCII Printers and Plotters."

Emulation of 328x Printers

An ASCII printer attached to the IBM 7171 may be used to emulate a 328x printer which is described in section "6.3 IBM 7171 Printer Support" on page 6-21. For example, the host may assign the device type as a 3286 printer. The 3270 Data Stream mode for controlling printers should be used. The SNA Character String mode is not supported.

Auxiliary Printers

Some ASCII display terminals allow an ASCII hardcopy device to be connected to the auxiliary printer port of the display terminal. This includes the IBM 3101 and the IBM Personal Computer running the IBM 3101 Emulation Program. Local printing is invoked by a key sequence issued from the terminal keyboard, which results in the contents of the screen being printed on the hardcopy device.

Similar functions may be available with non-IBM display terminals or personal computers.

2.4 Differences Between ASCII and IBM 3270 Devices

Note the following differences between ASCII printers and display terminals and IBM 3270 devices (EBCDIC):

- **ASCII Display Terminals** - ASCII display terminals, when used with the IBM 7171, are subject to the following restrictions:

  1. The character in the lower right hand corner of the screen is normally not displayed. On most ASCII terminals, any attempt to write a character in this position causes the screen to scroll up one line. Everything displayed

on the screen therefore appears one line higher than the IBM 7171 internal buffer indicates it is displayed, and the results of input are difficult to predict.

It is sometimes possible to suppress this scrolling, which is done on the IBM 3101 terminal with a Customer Setup Switch. When this is done on all terminals of this type, the Terminal Definition Table for the terminal type can be changed to set the X'0040' flag bit to enable display of this last character position on the screen. (See the FLAGS keyword in the TDT in "Chapter 4. Customizing IBM 7171 Tables")

2. The handling of nulls (X'00' characters) is somewhat different. On a standard 3270 terminal, the null character is suppressed for input from the terminal from a Read Modified command. The null character appears on the screen as a blank and may separate "two words" as the data is viewed by the user. If the null is suppressed on input the host reads "twowords" and the result is typically an error. The IBM 7171 reduces the possibility of error by translating imbedded nulls between words to blanks while continuing to suppress trailing nulls at the end of the field. This reduces the amount of data the host must process. To maintain consistency, the Delete key operates across lines in a multi-line field (where there is no attribute byte in either column 80 or column 1 of the next line). Applications which write multi-line fields with nulls and then read them back with Read Buffer would find that the arrangement of lines on the screen had been damaged by multi-line Delete. Correct operation of such applications can be restored by use of Zones Mode as described under "3.1.5 Setup Functions" on page 3-3. The improved null processing function of the IBM 7171 is an option that may be deactivated by the user.

3. The IBM 7171 supports features of the 3270 family as described in IBM programming and hardware manuals. It does not claim to emulate accidental hardware characteristics when a particular 3270 model is sent an incorrect data stream. In particular, it may be more restrictive than the real hardware in checking the validity of characters used to form buffer addresses, attribute bytes, or other control functions.

4. All terminals must communicate in full-duplex character mode. The IBM 7171 cannot support hardware features which are not available in this mode, such as highlighting on a 3101 Model 2x (in block mode). For more details on this subject, refer to "Chapter 3. Using ASCII Terminals on the IBM 7171."

- **ASCII Printers** - The results of including arbitrary bit combinations in files printed on ASCII printers cannot be predicted, since certain bit combinations may be interpreted as control characters. Thus, certain bit combinations can result in errors, especially in files containing machine programs of the VM/SP 'TEXT' file type. Printing of these files should therefore be avoided.

2.5 Customizing the IBM 7171

The IBM 7171 is ready to operate, provided its configuration fits the requirements of the user's installation. The following types of customization changes can be made:

1. **Changing Communication Lines** - For each IBM 7171 port address the line speed (or autobaud detection), parity selection, number of stop bits, terminal type, and other parameters are predefined but may be changed. For a complete definition and the default settings as delivered, refer to section "4.4.2 Ports Area Layout" on page 4-24. The procedure for changing these predefined parameters for each communication line is explained in "Chapter 4. Customizing IBM 7171 Tables."
2. **Generating Terminal Definition Tables** - Existing tables can be modified (customized) and new ones created for new terminal types. Detailed information on how to do this is contained in "Chapter 4. Customizing IBM 7171 Tables."
3. **Providing User Written RSCS Line Drivers** - For users with printers and plotters, information on line drivers is given in "Appendix D. Hints on Using ASCII Printers and Plotters."

2.6 Operating Environment of the IBM 7171

The IBM 7171 requires the following system configuration and programming support:

2.6.1 System Configuration

Host Processor: The IBM 7171 will support IBM 43xx or 308x processors running VM/SP release 3 or MVS release 3.8. The IBM 7171 must be attached to the host via a block multiplexer (BMPX) channel. Several IBM 7171's can be attached to a single BMPX channel at the same time. However, this could affect the performance of the IBM 7171 depending on what other I/O devices are attached to the same channel. It is recommended that the IBM 7171 not be put on the same channel as tape drives for performance reasons.

Up to 64 ASCII devices may be attached to a single IBM 7171. When more than 32 devices are attached, the IBM 7171 appears as two 3274-1D control units.

2.6.2 ASCII Display Terminal Requirements

To be attached to the IBM 7171, ASCII display terminals must meet the following minimum functional requirements. The ASCII display terminal must be capable of full duplex character mode operation, must be connected point to point, must use the Start/Stop control protocol, and must use 7 bit ASCII code. The terminal must perform the following functions upon receiving an appropriate character sequence from the control unit:

- Clear screen or clear to end-of-screen.
- Absolute cursor positioning.
- A character written to the screen should replace, not overstrike, the previous character in that position. APL mode is an exception, when overstrike characters must be formed.

In addition, the following display terminal features are desirable, and can be used if they are present:

- Transmitting cursor movement keys
- Transmitting program function keys
- Audible alarm ("beep")
- Controllable indicator to signal "insert" mode
- Erase to end-of-line (required for APL terminals).

2.7 Setup Requirements for the IBM 7171

For planning and installation information refer to the *IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide* listed in "Related Publications" on page iv. This reference gives site preparation and installation information including, connecting the host channel cables, setting the channel address of the IBM 7171, selecting the correct type of ASCII terminal cables, powering-up the IBM 7171, and trouble-shooting when the Ready indicator fails to light.

Once the system has been installed, it can be adapted to the needs of the installation as described earlier in section "2.5 Customizing the IBM 7171" on page 2-11.

Defining IBM 7171 and ASCII Terminals to the Host

Each IBM 7171 must be defined to the host system as one or two 3274-1D control units. An IBM 7171 with support for 32 or less ports will appear as one 3274-1D. One with more than 32 ports will appear as two 3274-1D control units.

The attached ASCII display terminals and printers appear to the host system as IBM 3278 or 3277 terminals and 328x printers. Each IBM 7171 must define its attached display terminals as the same type, either IBM 3278 or 3277.

2.7.1 Preparing the VM/SP System

An example of the necessary host entries for two IBM 7171's attached to the same host channel is shown in Figure 2-4. The first IBM 7171 has 64 ASCII devices attached and therefore requires two 3274's (two shared Unit Control Words). The second IBM 7171 has 32 ASCII devices attached and needs only one 3274.

```
**************************************************
*       IBM 7171 SYSTEM 1 WITH 64 ASCII DEVICES
RDEVICE ADDRESS=(c00,31),DEVTYPE=3278,MODEL=2
RDEVICE ADDRESS=(c1F,01),DEVTYPE=3286
RDEVICE ADDRESS=(c20,31),DEVTYPE=3278,MODEL=2
RDEVICE ADDRESS=(c3F,01),DEVTYPE=3286
*
**************************************************
*       IBM 7171 SYSTEM 2 WITH 32 ASCII DEVICES
RDEVICE ADDRESS=(c40,32),DEVTYPE=3278,MODEL=2
*
**************************************************
*       IBM 7171 SYSTEM 1 CONTROL UNIT
RCTLUNIT ADDRESS=c00,CUTYPE=3274,FEATURE=32-DEVICE
RCTLUNIT ADDRESS=c20,CUTYPE=3274,FEATURE=32-DEVICE
*
*       IBM 7171 SYSTEM 2 CONTROL UNIT
RCTLUNIT ADDRESS=c40,CUTYPE=3274,FEATURE=32-DEVICE
*
**************************************************
*       IBM 7171 SYSTEM 1 AND 2 CHANNEL
RCHANNEL ADDRESS=c,CHTYPE=BLKMPXR
*
**************************************************
```

Figure 2-4. VM/SP System Generation Information

where 'c' defines the channel address on the IBM 7171-host channel. For a detailed description of setting up channel addresses for the IBM 7171, refer to the document *IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide*.

2.7.2 Preparing the MVS System

An example of the device entries for one IBM 7171 attached to a host channel is shown in Figure 2-5.

```
**************************************************
*       IBM 7171 SYSTEM WITH 32 ASCII DEVICES
*
IODEVICE ADDRESS=(cA0,4),DEVTYPE=3278,MODEL=2,
      FEATURE=(AUDALRM,SELPEN)
IODEVICE ADDRESS=(cA4,2),DEVTYPE=3286,MODEL=2
IODEVICE ADDRESS=(cA6,26),DEVTYPE=3278,MODEL=2,
      FEATURE=(AUDALRM,SELPEN)
*
**************************************************
```

Figure 2-5. MVS System Generation Information

where 'c' is the block multiplexer channel and the devices are thirty 3278 model 2 terminals with AUDALRM and SELPEN features, and two 3286 model 2 printers.

2.8 System Start-up and Operation

Complete the installation procedures in the publication *IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide* to establish connection between the ASCII terminals and the IBM 7171.

2.8.1 Operating IBM 7171 Terminals

1. Insure that the terminal is ready. (Set up a valid baud rate, the correct number of stop bits ,the correct parity, etc.)

2. Activate the IBM 7171 power ON switch and wait about 10 seconds for the Diagnostics to complete. The Ready indicator light should come on.

3. Set the IBM 7171 ON LINE/OFF LINE mode switch to ON LINE.

4. Activate the power ON switch at the connected ASCII terminal.

5. Since autobaud is the default in all of the delivered terminal type tables, the host operating system logon display will not appear on the screen of the ASCII terminal when it is turned on. (However, if the default was changed to a fixed baud rate (autobaud off) and if the ASCII terminal type was predefined, the host logo would appear immediately.)

6. Press the key on the terminal (usually the RETURN key) that generates the carriage return (CR) character[6].

7. The following message will be displayed:

```
ENTER TERMINAL TYPE:
```

8. Enter the terminal type, for example, IBM3101. If an invalid terminal type is entered, and the Terminal Definition Tables delivered with the IBM 7171 are being used, the following display of valid terminal types will appear:

```
VALID TYPES ARE:

IBM3101[7]
TVI912     TVI920     TVI950     TVI950R
ADM31      ADM3A
VT100
DM1520     DM1521     DM3045
TYPETERM

ENTER TERMINAL TYPE:
```

9. After entering the correct terminal type, press ENTER.

10. The connected terminal should display the "logo" of the host operating system.

11. Logon to the host in the normal way.

    Note: If a valid terminal type is entered **other** than the type of terminal at which the message appears, the terminal may become inoperable. To correct this situation, execute the procedure described in "8.4.2 Terminal Type Specification Errors" on page 8-3.

12. If the host logon screen still does not appear, try hitting the CLEAR key. If the logon does not appear, the host system operator probably has not enabled the terminal line. Ask him to do this at the host system operator console.

---

6 For the IBM 3101 press the key with an arrow that points down and to the left. For the IBM PC running the IBM 3101 Emulation Program, press the key with an arrow pointing to the left. In general, the key, or key sequence must be pressed that generates the ASCII "CR" (X'0D') code character.

7 For an IBM PC running the IBM 3101 Emulation Program, IBM3101 must be used as the terminal type.

Chapter 3. Using ASCII Terminals on the IBM 7171

This chapter is for ASCII terminal users. It explains how the IBM 7171 emulates 3270 terminal functions for ASCII display terminals. It assumes that the user is familiar with the functions of a 3270 terminal. If this is not the case, the full complement of IBM 3270 Information Display System products is identified in the introductory publication *An Introduction to the IBM 3270 Information Display System,* GA27-2739. For more detailed 3270 information refer to the *IBM 3270 Information Display System Data Stream Programmer's Reference,* GA23-0059.

3.1 IBM 3270 Emulated and Extended Functions

The IBM 7171 not only emulates a 3270 terminal for its attached ASCII terminals, in some cases it extends these functions to make interactive text and program editing more effective.

3.1.1 Different Character Representation

There are two important EBCDIC characters which are not defined in ASCII and therefore usually cannot be found on the keyboard of an ASCII terminal:

```
- the "cent" symbol (¢)   and
- the "not" symbol  (¬)
```

Since these symbols can be important in certain program languages and editors, the IBM 7171 represents them with two less important ASCII characters:

```
- the "backslash" (\) is used for the "cent" (¢)
- the "circumflex accent" (^) is used for the "not" (¬)
```

Thus, the "backslash" (\) key and the "circumflex accent" (^) key on an ASCII terminal with the IBM 7171 are to be used in place of the "cent" (¢) and the "not" symbol(¬). Note that the "cent" (¢) and the "not" symbol(¬) appear on an ASCII terminal screen as "backslash" (\) and "circumflex accent" (^). Nevertheless, the hexadecimal notation of what appears as "backslash" (\) is X'4A', which is the EBCDIC code for the "cent" (¢), and of what appears as "circumflex accent" (^) is X'5F', which is the EBCDIC code for the "not" (¬).

3.1.2 Type-Ahead Capability

The 3270 hardware has a "keyboard lock" function which prevents the user from entering data before the application program is ready to receive it. In contrast, the IBM 7171 normally does not lock the keyboard of an ASCII terminal. It knows when the keyboard of a real 3270 would be locked and does not process user input until it receives an "unlock" message from the host. This occurs since ASCII terminals are connected through full-duplex asynchronous interface hardware, whereas 3270 terminals operate in half-duplex mode.

To indicate that the keyboard is logically locked, the cursor is moved to the lower right corner of the screen. In this mode, up to 64 characters of user input are held by the IBM 7171 in a "type-ahead" buffer. The characters that are typed do not appear on the screen because it has not yet been determined where they are to be placed. Since it is possible to type-ahead with any key, including, for example, ENTER and PFKs, it is quite possible that information is being typed which is logically associated with a screen image that the host computer has not yet written. When the host unlocks the keyboard, all the typed-ahead data appears on the screen where designated by the host. The resulting effect is identical if the user had simply waited for the host to unlock the keyboard and only then typed in the same information.

It is often useful to type-ahead multiple ENTER and PFKs, but each such key when processed by the IBM 7171 logically re-locks the keyboard and requires another host reaction and another unlock request before the data following it is processed.

3.1.3 Highlighted Fields

Highlighted fields are supported on terminals that allow individual characters to be highlighted, but which have no true attribute bytes. An example is the DEC VT-100 terminal, which supports the ANSI X3.64 standard for Set Graphic Rendition with parameter "1." The IBM 7171 supplied Terminal Definition Table for the DEC VT-100 provides this capability.

Highlighting is also supported on terminals that accept attribute bytes which govern the display of characters following them up to the next attribute byte. Terminals which accept attribute bytes, but which revert to some default display mode at the end of a line, are not supported.

Some terminals (e.g. TeleVideo 950) can handle both the former ("mode") and the latter ("attribute") highlighting. The two methods cannot be mixed within the same Terminal Definition Table.

3.1.4 Color Terminals

Color terminals which operate in ANSI or similar modes can be defined to emulate IBM 3279 "basic color." "Basic color" mode means that four different colors are assigned to the unprotected unhighlighted, unprotected highlighted, protected unhighlighted and protected highlighted fields. Selection of colors is table-driven and must be coded in the Terminal Definition Table of the terminal type. A keyboard sequence allows switching between alternate color selections, or between simple intensity highlighting and four-color modes. See "Alternate Display of

Attributes" on page 3-9. In this way, colors need only be used when they are helpful.

3.1.5 Setup Functions

Most of the extended functions provided by the IBM 7171 can be initialized by the ASCII terminal user through **Setup Functions**. The setup function can be executed from the user terminal by a predefined key-sequence. Refer to the Setup Functions table for a specific terminal type in "Appendix B. IBM 7171 Supplied Terminal Definition Tables" on page B-1.

Normally, the setup functions are defined in pairs; one setup function initiates the function and the other resets it to the previous or default definition. The setup function may represent a standard 3270 function, a function defined within the IBM 7171, or one determined by the user.

The function initiated by a setup function normally remains in effect until the corresponding reset setup function is issued. Some setup functions are executed by control keys defined for this purpose. Powering off the IBM 7171 resets the setup functions to the default states.

The following setup functions are available:

- Set Column Tab
- Delete Column Tab
- Set Left Margin
- Set Home Line
- Delete all Column Tabs, and reset Home Line and Left Margin
- Improved Null Processing
- 3270 Null Processing
- Zones Mode on
- Zones Mode off
- Reverse Enter/Newline Keys
- Restore Enter/Newline Keys
- Reverse Column and Field Tab Keys
- Restore Column and Field Tab Keys
- Alpha in Numeric-Only Field
- 3270 Numeric-only Fields
- 3278 Insert Mode
- 3277 Insert Mode
- APL Mode on
- APL Mode off
- ASCII Input in APL Mode
- Alternate Display of Attributes
- Primary Display of Attributes
- Suppress Pacing
- Restore Pacing
- Keyboard initiated Line Drop
- Return to ENTER TERMINAL TYPE Message
- Alternate Keyboard Arrangement
- Primary Keyboard Arrangement

In the following description the setup functions themselves appear in **bold** type, whereas the control keys are shown in CAPITAL letters.

Set Column Tab

The setup function **Set Column Tab** allows a column tab stop to be set at any column position on the screen. This provides a typewriter-style, column-oriented tabs ("column tabs") function (the regular 3270 field tabs function is also provided). Setup functions can be used to erase the single tab stop where the cursor is currently positioned (**Delete Column Tab**), or to delete all tab stops that have been defined previously **(Delete all Column Tabs, and reset Home Line and Left Margin)**

After setting of the column tab stops, the COLUMN TAB key can be used to move the cursor from its current position right to the next column tab stop, and the COLUMN BACKTAB key can be used to move the cursor to the previous column tab stop.

Delete Column Tab

The setup function **Delete Column Tab** is used to erase the single tab stop where the cursor is currently positioned.

Set Left Margin

The setup function **Set Left Margin** sets the left margin to the current cursor column position. The column need not be a column tab stop. Any later use of the NEWLINE key functions the same way as described under "Automatic Indentation." The setup function **Delete All Column Tabs, and reset Home Line and Left Margin** resets the left margin to the leftmost position on the screen.

Automatic Indentation

This special feature, although not a setup function, is related to setting up column tabs and the left margin. The NEWLINE key normally moves the cursor to the first unprotected character on the next line. The IBM 7171 allows a generalization of the "newline function" in that it allows the cursor to be moved left to some other column in the next line on the screen. That means that the "left margin" is not fixed at the physical left end of the screen, but can be changed. After some column tab stops are defined on the screen, the INDENT and UNDENT keys are used.

The INDENT key is used to move the cursor to the next tab stop to the right of its current position, exactly as the COLUMN TAB key does. In addition, however, this key will set that column to be the left margin for the newline function. If the INDENT key is pressed and there are no more column tab stops to the right of the current cursor location, the terminal will respond with an audible alarm ("beep"). Whenever the cursor is in a column to the right of the left margin and the NEWLINE key is pressed, the cursor will move to the left margin on the next line that has unprotected characters. If the next line has a protected character at the left margin, the cursor will move to the first unprotected character to the right of the newline left margin. If the cursor was already in that column, it will just move down one line. If, however, the cursor was positioned in any column to the left of the newline left margin, it will be moved to the first unprotected character of the next line that has unprotected characters.

The UNDENT key is used to move the cursor and the left margin to the next column tab stop on the left of the current cursor position. If the UNDENT key is pressed and there are no more column tab stops on the left of the current cursor location, the cursor will be moved to the physical beginning of the current line.

Set Home Line

The setup function **Set Home Line** can be used to redefine the location of the home position ("home line") of the cursor. The line that the cursor is currently on becomes the home line. Pressing the HOME key will then move the cursor to the first unprotected position on that line. The setup function **Delete All Column Tabs, and reset Home Line and Left Margin** resets this redefined home position.

Delete All Column Tabs, and Reset Home Line and Left Margin

The setup function **Delete All Column Tabs, and reset Home Line and Left Margin** clears all column tab stops, resets the home line to the first line with unprotected characters, and resets the left margin to the leftmost position on the screen.

Improved Null Processing

Both the null (X'00') and the blank character (space) appear on the terminal screen as a blank space. However, a real 3270 terminal has its own way of treating nulls and blanks, which can lead to confusion, since the user cannot tell by looking at the screen which spaces are nulls and which are blanks. **Improved Null Processing** treats nulls and blanks in a consistent way, thus producing a "What you see is what you get" effect and relieving the user of the need to distinguish between these two characters.

1. On a real 3270, when fields which have been modified by keyboard entry are read by the application, nulls are not transmitted to the host, even when they are imbedded among visible characters in the field. Thus, the host may "see" the field differently than the user, as in the following example where nulls occupy the space between "...BUF" and "A NEW...."

   ```
   Appearance on screen: GET   INDCB,BUF           A NEW RECORD
   Sent to host:         GET   INDCB,BUFA NEW RECORD
   ```

   Using the setup function **Improved Null Processing**, which is the default, the IBM 7171 translates nulls between characters into blanks when reading from the screen. The above input line would thus produce the result normally expected:

   ```
   Sent to host:         GET   INDCB,BUF           A NEW RECORD
   ```

   Trailing nulls on the line are still suppressed, because they are not significant, and removing them reduces the number of characters which the IBM 7171 must process.

2. On a real 3270, insert mode operates only in fields padded with nulls. In the following example, the symbol '|' represents the physical location of the attribute byte which ends the previous field and begins a new field (on the screen, the attribute byte displays as a blank).

   ```
       |The terminal shold do what you expect it to do.     |
                        /
   ```

In the above line, the missing "u" could only be inserted before the character above the slash if the characters at the end of the line were nulls. If the characters between the period ending the sentence and the physical end of field were all blanks, an attempt to insert on a real 3270 will produce an error condition, which locks the keyboard and must be cleared with the RESET key. Since nulls appear as blanks, the user cannot tell if insertion is legal until he tries to insert a character.

The IBM 7171 will allow either blanks or nulls to be pushed off the end of a field.

3270 Null Processing

The setup function **3270 Null Processing** allows regular 3270-type null processing to be used. Blanks may still be pushed off the end of a field when in insert mode, however.

Zones Mode On

In the 3270 mode of operation, attribute bytes define tab stops ("field tabs") which not only delimit fields, but also delimit the scope of the INSERT and DELETE functions. Since column tabs provide an alternate tabbing mechanism without the use of attribute bytes, a similar mode of operation has been introduced which is called "zones mode." A "zone" is a part of a line on the screen which is delimited by two column tab stops. If there is no column tab stop on either the first or last physical position on the line, the area between the beginning of the line and the first column tab stop, or between the last column tab stop and the end of the line, also represents a zone. When the user is in "zones mode" and presses the INSERT key, the character to be inserted is placed at the current cursor position. The original character which was at the current cursor position and subsequent characters are "rippled" one position to the right. The "ripple" of characters for INSERT in zones mode will stop at any column tab stop which is preceded by three or more blanks (or nulls). When the user presses the DELETE key in zones mode, the character at the current cursor position is deleted and subsequent characters are "rippled" one position to the left . The "ripple" of characters for DELETE in zones mode will stop at any column tab stop which is preceded by two blanks (or nulls).

The setup function **Zones Mode On** allows the user to select this mode of operation.

Zones Mode Off

The default setting **Zones Mode Off** causes the zones mode operation, previously selected by the setup function **Zones Mode On**, to be terminated. The standard 3270 field tab oriented mode of operation is re-invoked.

Reverse ENTER/NEWLINE Keys

The setup function **Reverse ENTER/NEWLINE Keys** allows the user to reverse (exchange) the ENTER and NEWLINE keys. The ENTER and NEWLINE keys for the supported terminal types are defined in "Appendix B. IBM 7171 Supplied Terminal Definition Tables." In Appendix B refer to the table "Control and Cursor Movement Keys" for the specific terminal type. To reverse these keys, type in the key sequence as shown in the table "Setup Functions" in Appendix B for the specific terminal type.

Restore ENTER/NEWLINE Keys

The setup function **Restore ENTER/NEWLINE Keys** enables the user to restore the previous setting of the ENTER and NEWLINE keys before the setup function "Reverse ENTER/NEWLINE Keys" was issued.

Reverse Column and Field Tab Keys

The setup function **Reverse Column and Field Tab Keys** allows the user to reverse (exchange) the COLUMN TAB with the FIELD TAB key, and the COLUMN BACKTAB with the FIELD BACKTAB key.

Restore Column and Field Tab Keys

The setup function **Restore Column and Field Tab Keys** is used to restore the previous setting of the COLUMN TAB, FIELD TAB, COLUMN BACKTAB and FIELD BACKTAB keys before the setup function **Reverse Column and Field Tab Keys** was issued.

Alphanumeric in Numeric-Only Field

The setup function **Alphanumeric in Numeric-Only Field** allows the entry of any character (including non-numeric characters) into numeric fields.

3270 Numeric-only Fields

The setup function **3270 Numeric-only Fields** is used to restore the original numeric-only restriction to numeric fields.

3278 Insert Mode

The setup function **3278 Insert Mode** provides the 3278 approach of terminating insert mode when an attention-generating key ("ENTER," "PF" or "PA") is pressed.

3277 Insert Mode

The setup function **3277 Insert Mode** returns the terminal to the default setting, where only the insert toggle key (In Appendix B see "Toggle Insert Mode" in the table "Control and Cursor Movement Keys" for the specific terminal type) ends insert mode.

APL Character Set - General Information

When using APL, the special APL character set must be displayed on the screen. Some terminals can display both the normal (ASCII) character set and the APL character set at the same time. Some terminals can only display one set at a time. The IBM 7171 will allow the use of either kind of terminal. When an APL character is sent to a terminal which is not in APL display mode, or an ASCII character is sent to a terminal which is currently only able to display APL, the character is displayed as a special invalid token character (normally a blot or a colon).

The keys on the keyboard of an APL terminal usually include both sets of characters. The keys always send the same code to the computer whether the terminal is in APL mode or not. If the terminal is able to display both APL and non-APL characters at the same time, some keys are needed which mean "now treat everything typed as APL characters" and "now treat everything as normal (lowercase) letters and symbols." With such a facility, it is possible to edit lowercase characters into APL variables or to edit APL examples into a SCRIPT data set.

Certain APL characters are compounds that are formed by overtyping one APL symbol with another. The terminal support must be able to form such compound characters from the keyboard and to display them on the screen. Composite characters are formed by overstriking. The rule is: if two simple (non-composite) characters are overstruck, and if both together form a valid composite, then the composite is formed and displayed on the screen. If the two characters do not form a valid composite, or if the screen position already has a composite character, then the keyboard input replaces the previous character.

Much of the APL processing is automatic. However, there are three special setup functions which are defined to enter and exit APL mode:

APL Mode On

The setup function **APL Mode On** turns APL on and treats all subsequent input from the terminal as APL characters. If the terminal is only able to display one kind of character set at a time, receipt of this signal will also cause the screen to be rewritten with lowercase interpreted as uppercase and non-APL symbols changed to the APL illegal character token. When both character sets can be displayed at the same time, the screen is not automatically refreshed by this key.

APL Mode Off

The setup function **APL Mode Off** turns APL mode off and treats all subsequent input from the terminal as normal ASCII characters. If the terminal is only able to display one character set at a time, activating this function causes the screen to be redisplayed in upper case and with APL special characters converted to the normal ASCII illegal character token.

ASCII Input in APL Mode

The setup function **ASCII Input in APL Mode** can be used when the terminal is in APL mode. It leaves the display in APL mode but treats subsequent input from the keyboard as ASCII. It is used to enter lowercase letters into APL variables or when editing mixtures of APL and text. This key is normally available only on terminals which can display both APL and ASCII at the same time. It differs from **APL MODE Off** in that the APL characters on the screen continue to be displayed correctly (if **APL MODE Off** is selected and APL characters are moved by the host or by insert mode or DEL, then they are converted to the illegal token). The keyboard can be shifted back to APL by typing in the key sequence for the setup function **APL Mode On** again.

Alternate Display of Attributes

The setup function **Alternate Display of Attributes** selects an alternate display mode, if one is defined. There are two setup functions which select primary or alternate display of attributes. On terminals with highlighting or color, there can be two alternate methods of displaying highlighted/normal and protected/unprotected fields.

Primary Display of Attributes

The setup function **Primary Display of Attributes** restores the primary mode of display, which is the default.

Suppress Pacing

The setup function **Suppress Pacing** controls whether pacing of characters is defined for the terminal being used. If the appropriate key sequence for **Suppress Pacing** is typed in, the local reset and control functions "Pacing Start" and "Pacing Stop" (normally XON/XOFF) will be disabled and the defined pacing characters will be treated as normal control characters.

Restore Pacing

The setup function **Restore Pacing** allows the pacing characters to be used to control output from the IBM 7171; e.g., the local reset and control functions "Pacing Start" and "Pacing Stop" will be enabled again. See section "3.1.6 Local Reset and Control Facilities."

Keyboard Initiated Line Drop

The setup function **Keyboard Initiated Line Drop** performs a keyboard initiated disconnect. The IBM 7171 transmits a disconnect string to reset the terminal to an appropriate state and if appropriate, then drops the phone line.

Return to ENTER TERMINAL TYPE Message

The setup function **Return to ENTER TERMINAL TYPE Message** returns the terminal to the "ENTER TERMINAL TYPE:" prompt to allow selection of an alternate table for this terminal. The communications link is not dropped. If a host session is in progress it is disconnected.

Alternate Keyboard Arrangement

The setup function **Alternate Keyboard Arrangement** logically rearranges the keyboard to match the Dvorak keyboard arrangement. The Dvorak keyboard arrangement is a keyboard layout designed to optimize letter placement so that faster typing is possible. Figure 3-1 gives an example of this arrangement for the IBM 3101. The default keyboard setting is restored by the setup function "Primary Keyboard Arrangement."

```
lower:  1234567890-=        upper: °!@#$%¬&*()_+
        /,.pyfgcrl[¢                ?<>PYFGCRL]|
        aoeuidhtns'                 AOEUIDHTNS"
        ;qjkxbmwvz                  :QJKXBMWVZ
```

Figure 3-1. Example of Dvorak Keyboard Arrangement for IBM 3101

Primary Keyboard Arrangement

The setup function **Primary Keyboard Arrangement** restores the use of the normal QWERTY keyboard arrangement.

3.1.6 Local Reset and Control Facilities

In the IBM 7171 there are six **Local Reset and Control Functions**, which are requests that are routed directly to the IBM 7171. They are not passed to the host but provide control over the transmission and clearing of data to and from the terminal. The functions are:

- **MASTER RESET** clears all pending character or transmission error indicators, purges all typed ahead characters, re-transmits the terminal initialization sequence, rewrites the screen image from the screen image buffer and exits from insert mode. This function solves most user problems and is a convenient "panic button."

- **CHARACTER ERROR RESET** clears the error condition detected indicator after a parity, framing, break interrupt, or overrun error from the terminal and makes further terminal input possible.

- **KEYBOARD UNLOCK** clears the logical keyboard lock and allows further input to be processed. If input has been typed ahead, it will be processed immediately. Otherwise, input will be received from the terminal. This is similar to RESET on a real IBM 3278.

- **TYPE-AHEAD PURGE** flushes the contents of the type-ahead buffer. This gives the user the ability to correct errors which have been typed in but not yet processed, and it is recommended if the user changes his mind or needs to perform some other function immediately.

- **PACING START** is normally XON. When this character is received and pacing is enabled (the setup function **Restore Pacing** is in effect), transmission to the terminal will be resumed. If transmission is not currently halted by PACING

STOP, but pacing is enabled, the audible alarm ("beep") is sent to the terminal and this character is ignored. Pacing may be disabled by the setup function **Suppress Pacing** (see "Suppress Pacing") and reenabled again by the setup function **Restore Pacing**.

- **PACING STOP** is normally XOFF. When this character is received, and if pacing is enabled (the setup function **Restore Pacing** is in effect), transmission to the terminal will stop until the PACING START character (see above) is received. If transmission is already halted and pacing is enabled, this character is ignored. Pacing may be disabled by the setup function **Suppress Pacing** (see "Suppress Pacing") and reenabled again by the setup function **Restore Pacing**.

3.2 Error Situations

The following gives a description of some more important error situations which a user may encounter when operating an ASCII terminal.

3.2.1 Errors on Communication Lines

Errors can occur in data transmitted between the ASCII terminal and the IBM 7171. When the connection is made using normal phone lines, the error can be caused by background noise produced by the telephone equipment. For locally connected terminals such errors are less common, but may still be caused by electrical interference.

When a transmission error is detected on input data (bad parity, framing error, or overrun) the terminal is marked in an input-error state. Every subsequent character received is ignored and a BEL character is echoed back to the terminal causing an audible alarm ("beep") to inform the user of the error state. The terminal user must explicitly acknowledge the error by entering the Character Error Reset function (see "3.1.6 Local Reset and Control Facilities" on page 3-10). After receiving the key sequence for this function, the IBM 7171 clears the error indication for the terminal and normal input can continue.

The BREAK key on the keyboard transmits a sequence which the IBM 7171 will regard as a character transmission error. Therefore, if a terminal user accidentally hits the BREAK key, he must correct the error by typing in the key sequence of the Character Error Reset function. Should connection with the terminal be lost due to modem or phone line errors, the line will be disabled and reenabled to permit the user to dial back in and logon.

3.2.2 Trying to Change Protected Data

There are certain parts of the screen into which the user cannot type data. For example, in a full-screen editor there are usually sections in the top line of the screen which contain status information which cannot be modified. There are also field separators called "attribute bytes" that separate the various fields of each screen line, and the user cannot modify or delete these field separator locations. Finally, when the terminal is in insert mode, and the cursor is positioned in a field which is full (the last character in the field is neither 'null' nor 'blank'), then no additional characters can be added to the field.

In all three cases, the IBM 7171 responds to an illegal attempt to change protected data, or to add to a full field, with an audible alarm ("beep") and by leaving the screen unchanged.

Once the problem is corrected by moving the cursor out of the protected field, the IBM 7171 will continue normal operation. It is not necessary to explicitly press a "reset" key in any of these circumstances as would be required on a "real" 3270. Note, however, that the IBM 3101 Model 2x terminals will generate their own Lock/Reset condition when one of the block-mode-only keys is pressed.

3.2.3 Pressing the Wrong Key

If any control request is entered out of context, the request will be ignored and the ASCII terminal will respond with an audible alarm ("beep"). For example, a Character Error Reset function issued when there is no error situation present, or a Keyboard Unlock function issued when the keyboard is already unlocked will generate a "beep." Similarly, any attempt to use an undefined CTRL key (ALT key on the IBM 3101), ESC or PFK sequence will also generate a "beep."

Once the problem is corrected by pressing the correct key, the IBM 7171 will continue normal operation. It is not necessary to explicitly press a "reset" key as would be true on a "real" 3270. Note, however, that the IBM 3101 Model 2x terminals will generate their own Lock/Reset condition when one of the block-mode-only keys is pressed.

3.2.4 Errors on Output Data Stream

Errors may also be introduced into the output data stream transmitted from the host to the ASCII terminal. These errors appear as "garbage" characters somewhere on the screen. When the user suspects that the contents of the screen is in error, he can request that the screen image be retransmitted. The "Redisplay" control function (see "Control and Cursor Movement Keys" for the specific terminal) is used to request such retransmission. Note, however, that this function will not take effect until the keyboard is unlocked.

3.2.5 Full Type-Ahead Buffer

Within the IBM 7171 each ASCII terminal has a 64 character buffer for type-ahead. When this buffer is filled and additional characters are typed, then the IBM 7171 has no place to store them and responds with an audible alarm ("beep"). Since some data has been lost there is no automatic way to continue normal processing. The user could simply wait for the entire contents of the type-ahead buffer to be processed. When the cursor moves out of the lower right corner, then the buffer is empty and the user can see which characters were received and which were lost. Then the Character Error Reset function (see "3.1.6 Local Reset and Control Facilities" on page 3-10) will reset the error state and the user can enter additional data.

3.2.6 Manually Unlocking the Keyboard

There are times when the keyboard is logically locked, but the terminal user wants to manually unlock it. This is the case, for example, when he wants to interrupt an executing program with a program attention. On a 3270 this function is performed by the RESET key. For an ASCII terminal on the IBM 7171, there exists a Keyboard Unlock function, which can be found under "Local Reset and Control Functions" in this chapter. If there are any characters in the type-ahead buffer when the Keyboard Unlock function is entered, they will immediately be processed and may appear on the screen in the current cursor position. Should an attention generating key (ENTER or a PFK) be present in the type-ahead buffer it will cause the keyboard to lock again. For example, if while the keyboard is locked the user has typed

```
abc<ENTER>def
```

and now a Keyboard Unlock function is typed, the letters "abc" will appear on the screen and an ENTER notification will be sent to the host. The keyboard is logically locked by this ENTER and the "def" will remain in the type-ahead buffer. A second Keyboard Unlock function can be entered to unlock the keyboard again.

3.2.7 Master Reset

For simplicity reasons, a Master Reset function has been defined, which combines all the possible reset services in a single function. When issued, it purges the type-ahead buffer, clears error indicators, ends insert mode and rewrites the screen. The Master Reset function is much easier to remember than the group of other reset functions which it can replace. When a Master Reset is done, any problem in the terminal communications should have been cleared up. If subsequent data continues to generate an audible alarm ("beep"), the cursor is probably in a protected field (refer to "3.2.2 Trying to Change Protected Data" on page 3-11). See "Local Reset and Control Functions" for the specific terminal in "Appendix B. IBM 7171 Supplied Terminal Definition Tables" for the key sequence to be issued to generate a Master Reset.

3.2.8 Other Errors

For more information describing IBM 7171 error conditions refer to "Chapter 5. IBM 7171 I/O Interface to Terminals" for terminal errors, "Chapter 6. IBM 7171 I/O Interface to the Host System" for channel errors and "Chapter 8. Problem Determination" for general system errors.

3.3 Functions of Specific ASCII Terminals

"Appendix B. IBM 7171 Supplied Terminal Definition Tables" shows how ASCII code character sequences are related to the IBM 7171 and host processor functions. Most of this information is given in tables, which contain the key sequences for the following ASCII display terminals:

- IBM 3101 (Model 1 and 2 in character mode)
- IBM Personal Computer running in 3101 Emulation Mode

- TeleVideo TVI 912, 920, 950
- Lear Siegler ADM 3A, 31
- Digital Equipment VT 100
- Datamedia DM 1520, 1521, 3045.

In addition, there are tables which describe basic functions of a typical ASCII typewriter terminal without specifying any brand or model. See section "B.15 TYPETERM Typewriter Terminal" on page B-53 for more information.

Chapter 4. Customizing IBM 7171 Tables

4.1 Brief Overview

In order for the IBM 7171 to communicate with attached ASCII devices, the characteristics of each individual device and of each communication line must be specified. If the device type is not currently supported or the user wishes to change the communication line characteristics, the information must be supplied by the user.

Within the IBM 7171, terminal information is organized into two main functional areas. These are:

1. Terminal Definition Tables

2. Ports Configuration Data

4.1.1 Terminal Definition Table

A Terminal Definition Table (TDT) is a list of control character sequences that determine how ASCII terminals are to function. Each functionally different type of terminal must have its own TDT.

The IBM 7171 contains resident TDTs for 13 different terminal types. These are:

IBM3101 For the IBM 3101 terminal (Also used by the IBM PC in 3101 emulation mode)

TVI912 For the Televideo 912

TVI920 For the Televideo 920

TVI950 For the Televideo 950

TVI950R For the Televideo 950 in modified display mode

ADM31 For the Lear Siegler ADM 31

ADM3A For the Lear Siegler ADM 3A

VT100 For the Digital Equipment VT100 family

DM1520 For the Datamedia 1520

DM1521 For the Datamedia 1521

DM3045 For the Datamedia 3000 family

TYPETERM For generic typewriter terminals

HARDCOPY For generic hardcopy terminals.

All of the TDTs for these terminals are stored in Read-Only Memory (ROM). However, additional TDTs may be added and stored in Non-Volatile RAM (NV-RAM). This information will not be erased when the control unit is powered down.

Also stored in the ROM is a list of addresses to the standard strings, TDTs, and translate tables. The user is advised to use the addresses found in this list for all interaction. The list is found in section "4.5.4 ROM Pointers to Internal Information" on page 4-47 and in "Appendix F. ROM Data Base Organization."

4.1.2 Ports Configuration Data

For successful operation, certain Ports Configuration Data must be set for each active port. As explained below, default values are provided as well as means to change these values. The following options may be set for each Port:

Option **Argument**

#R The number of rows on the screen. A 0 defaults to 24 rows.

#C Number of columns on the screen. A 0 defaults to 80 columns.

BAUD Baud rates are supported two ways, autobaud, and non-autobaud:

1. Autobaud rates are supported by setting the baud rate equal to zero. Autobaud is the default. The baud rates supported in autobaud mode are 300, 600, 1200, 1800, 2400, 3600, 4800, 9600 and 19200 baud.

2. Non-autobaud rates that are supported are 50, 75, 110, 134, 150, 2000, and 7200 baud.

FLAG Flag word. There are two bytes of flags. The meanings of these flags are as follows:

Byte 1 Flags.

Bits ***Meaning***

20 Stick bit. Off or zero = Parity bit is interpreted as even or odd, On or one = Parity bit is interpreted as mark or space.

10 Parity. Off or zero = Odd parity (or if Stick bit is on always a 1 - Mark parity), On or one = Even parity (or Space parity if Stick bit is on).

08 Parity Enable. Off or zero = Parity Disabled, On or one = Parity Enabled

04 Stop Bits. Off or zero = 1 stop bit, On or one = 2 stop bits

02-01 Data Bits. 00 = 5 data bits, 01 = 6 data bits, 10 = 7 data bits, 11 = 8 data bits.

Byte 2 Flags.

Bits ***Meaning***

02-01 Type of connection. 00 = Let 7171 default, 01 = Switched Network (Telephone Line), 10 = Leased Line, 11 = Direct Connect

If both bits are zero, the 7171 will automatically determine the type of line that is attached.

TYPE This 8-byte field specifies which terminal type to assign to this port at line-connect time. The terminal type is the pointer to a valid Terminal Definition Table. If this field is zeroes then the terminal will prompt with the 'ENTER TERMINAL TYPE' message.

This chapter describes two different methods of accomplishing the required table and communication line changes. It also explains the organization and layout of the NV-RAM which is accessible to the user.

4.2 How to Specify Terminal Information.

There are two ways for a user to specify terminal information. They are:

- Via an automated program
- Via the Maintenance Facility.

Automated Program

An interface exists for a user created program to automatically access and store data into memory. For complete details, refer to "Appendix E. Interface for User Supplied Table Modification Program."

An implementation of this interface is supplied with the IBM 7171 and is described in "Appendix C. IBM 7171 Support Utility for Modifying Terminal Tables."

Manual Table Entry

The second way to enter terminal information is by using the Maintenance Facility. The Maintenance Facility allows the user to directly modify RAM and NV-RAM. For complete instructions for use of the Maintenance Facility, refer to "Chapter 9. Special Maintenance Facility and System Messages." The user should read and understand this chapter fully before entering the terminal information.

Note: Anyone wishing to create his own table generating program, or wishing to manually create or modify any existing terminal table MUST have working knowledge of the following concepts, which apply to IBM 7171 memory:

1. Memory addresses are given as a SEGMENT and an OFFSET. For the NV-RAM, the segment will always be X'DC00'. When an address is given by itself, that segment value can be assumed.

2. Addresses and word values are stored BYTE REVERSED in memory. That is if an address or word X'1234' is known to exist at memory location X'5678' the two bytes will be reversed. That is, the byte at address X'5678' will be X'34', and the byte at address X'5679' will be X'12'.

3. Table addressing in the 7171 frequently uses pointers. In this context the term "pointer" has a unique definition:

```
If bit 15 [the Most Significant Bit (MSB)] = 0

THEN
  The contents of the pointer are a relative
  (to segment X'DC00') address (max of 32k).
ELSE
  The contents of the pointer are an index that is
   converted into a displacement into the beginning of ROM
   (relative to segment X'DC00', offset X'4000').
  Bits 6-0 are the ROM index and allow addressing
   of the 128 entries (256 bytes).
  Thus pointer (index) X'8000' will address the
   ROM location the address of which is at X'4000'.
```

```
See Figure 4-9 on page 4-47 for indices and corresponding
ROM addresses.
The location in ROM contains the relative address of
the named table or character string.
```

Warning: If the Terminal Information is entered inaccurately, unexpected and catastrophic events may occur.

4.3 Terminal Definition Table Information.

The following information is necessary for describing a terminal type:

- Terminal Name
- Terminal Header Information
- Keyboard Input Strings
  - Immediate (Reset Character String)
  - Non-immediate (Input parse table)
- Terminal output
  - Control Sequence Strings
  - Highlighting Strings
- Translate tables

This section describes the terminal data required. Section "4.4 Organization of NV-RAM" on page 4-22 describes the detailed layout of this data in the 7171 storage.

4.3.1 Terminal Name

Each terminal must be given a name which may consist of up to eight characters. The name is associated with a pointer to a terminal header.

4.3.2 Terminal Header Information

The terminal header consists of terminal specific information and pointers to tables which contain terminal specific information. The terminal specific information defines such things as what the cursor origin is, if the terminal supports APL, if and what type of highlighting is supported, delays which may be required, etc. The pointers associate with this terminal the following:

- Input parse table
- Reset characters
- Output strings
- Graphics and Highlighting strings
- Translate tables

4.3.3 Keyboard Input

The IBM 7171 performs various functions or commands in response to keyboard entries. These keyboard entries can be categorized as immediate or non-immediate. Immediate keyboard entries (functions) are always processed as they occur. These are called Reset Keyboard Sequences and are described in section "Defining a Reset Keyboard Character Sequence" on page 4-11.

Non-immediate keyboard entries are normally processed as they occur. However, some of these functions cause the host to inhibit input from the keyboard (i.e. keyboard is locked) until command processing has been completed. If the keyboard is locked, non-immediate commands are inserted into a typeahead buffer for the appropriate terminal. These commands are then executed from the typeahead buffer in sequential order (first in, first out). Section "Non-Immediate Functions" describes the Non-Immediate keyboard functions.

Non-Immediate Functions

Non-Immediate keyboard entries are either graphic display characters or ASCII control strings which perform appropriate terminal commands. Normal graphic characters usually have ASCII codes that lie between hex'20' and hex'7E' inclusive. When these graphics are sent from a terminal, the character is placed into the screen buffer at the current cursor position and is also sent, or 'echoed', back to the terminal. No host interaction is involved. Figure 4-1 lists the terminal commands supported. Details of each command are described in the following sections.

```
                     FUNCTIONS SUPPORTED by the IBM 7171

NON-IMMEDIATE FUNCTIONS
-----------------------

   CURSOR             HOST                  FULL-SCREEN      OTHER
   MOVEMENT           INTERACTION           EDITING          KEY
   KEYS               KEYS                  COMMANDS         COMMANDS
   --------           -----------           ---------        --------

  CURSOR RIGHT        CLEAR                 INSERT TOGGLE    REDISPLAY
  CURSOR LEFT         ENTER                 DELETE CHARACTER LOCAL PRINT
  CURSOR UP           TEST REQUEST          ERASE EOF        DUP
  CURSOR DOWN         CURSOR SELECT         ERASE INPUT      FM
  FIELD TAB                                                  (See Setup
  FIELD BACKTAB                                              Functions)
  COLUMN TAB
  COLUMN BACKTAB
  NEWLINE
  HOME                PA KEYS
  UNDENT              PF KEYS
  INDENT
```

Figure 4-1. Functions Supported by the IBM 7171

Cursor Movement Keys: The cursor can be moved to other screen positions with the cursor movement keys (usually marked with arrows). Each depression of a cursor movement key moves the cursor one position in the direction indicated. Cursor movement wraps. If the cursor is moved off the right side of the screen, it reappears at the left side of the screen, one line down and vice-versa. If it is moved off the top of the screen, it re-appears at the bottom and vice-versa. Each of the cursor movement keys is described below:

Move Cursor Right
The Cursor Right command indicates one depression of the cursor right key(s) and will cause the IBM 7171 to transmit to the terminal a cursor reposition command to the new cursor position with screen wrap considered.

Move Cursor Left
The Cursor Left command indicates one depression of the cursor left key(s) and will cause the IBM 7171 to transmit to the terminal a cursor reposition command to the new cursor position with screen wrap considered.

Move Cursor Up
The Cursor Up command indicates one depression of the cursor up key(s) and will cause the IBM 7171 to transmit to the terminal a cursor reposition command to the new cursor position with screen wrap considered.

Move Cursor Down
The Cursor Down command indicates one depression of the cursor down key(s) and will cause the IBM 7171 to transmit to the terminal a cursor reposition command to the new cursor position with screen wrap considered.

Field Tab Key
The Field Tab sequence causes the IBM 7171 to send to the terminal a cursor reposition command. The cursor is repositioned at the first character position of the next unprotected field on the screen. The search for a field begins at the current cursor position, scans left to right on a line, then wraps to the beginning of the next line and continues. If there is not a new field before the end of the screen, the search wraps to the top of the screen and continues. If there are no unprotected fields on the screen, the cursor is not repositioned.

Field Backtab Key
The Field Backtab Key causes the IBM 7171 to reposition the cursor at the beginning of the current unprotected field on the screen unless the cursor is already there. If the cursor is at the beginning of an unprotected field, then the cursor is re-positioned to the beginning of the previous unprotected field. The method of 'scanning' the screen is opposite that of the field tab.

Home Key
The Home sequence moves the cursor to the first unprotected character position on the display screen.

Newline Key
The Newline sequence moves the cursor to the first line with unprotected data.

Indent Key
The Indent sequence repositions the cursor one tab stop to the right of its current position and sets the Newline left margin to the new cursor position. If

you press the Indent key and there are no more column tab stops to the right of the current cursor position the terminal will respond with with an audible alarm ("beep").

Operation of the Newline key is now dependent on the position of the cursor and whether there are protected or unprotected characters on the lines below the cursor. If the cursor is to the left of the newline left margin, pressing the Newline key will move the cursor to the first unprotected character of the next line that has unprotected characters. If the cursor is to the right of the newline left margin, pressing the Newline key will move the cursor to the newline left margin on the on the next line which has unprotected characters or if there are protected characters on the next line (occupying the newline left margin) the cursor will be moved to the first unprotected character to the right of the newline left margin.

Undent Key

The Undent sequence repositions the cursor one tab stop to the left of its current position and sets the Newline left margin to the new cursor position. If you press the Undent key and there is no column tab stop on the left of the current location, the cursor will be moved to the physical beginning of the current line.

Column Tab key

The Column Tab sequence repositions the cursor at the next tab stop to the right, without regard for protected fields. The tab stops may be set by the application program or by the terminal user.

Column Backtab key

The Column Backtab sequence repositions the cursor at the next tab stop to the left, without regard for protected fields. The tab stops may be set by the application program or by the terminal user.

Editing Keys

Enter/Exit Insert Mode Toggle

The Insert Mode sequence puts the terminal into the Insert Character mode. If the cursor is located in an unprotected data field having a null character either in the location identified by the cursor or in any location beyond the cursor, operation of any alphanumeric key causes that character to be entered at the cursor position. The character formerly occupying the cursor location and all remaining characters within the field (except nulls) will be shifted one character position to the right.

Repeating the Insert Mode sequence, releases the terminal from Insert Character mode. All characters entered into unprotected fields overtype data already in those particular screen buffer positions.

Delete a Character

The Delete sequence indicates that the character in the current cursor position is to be deleted and everything else in that field moves one character to the left. If the field 'wraps' from the end of the current line to the beginning of the next, the leftward movement of characters also wraps.

Erase to End-of-Field (End-of-Line)

The Erase to End-of-Field sequence causes a command to be sent back to that terminal which erases all data at and beyond the current cursor position. If the cursor is located in an unprotected field, all character positions from the current cursor position to the end of the line are deleted. There is no cursor re-positioning as a result of this sequence.

Erase Input

The Erase Input sequence causes a command back to be sent to that terminal which changes all unprotected data areas in the screen buffer to nulls, and re-positions the cursor to the first unprotected character location on the screen.

Host Attention Keys: All of the keys listed in this section signal the host CPU via a 7171 generated Attention Identification (AID) byte that input has been completed or to request front the host a service. The contents of this byte identify the cause of the interrupt (i.e., PA2, PF 11, CLEAR, etc.). Described below are host attention keys.

Clear

When the IBM 7171 receives the Clear sequence from a terminal, it forms the 3270 command string and responds to the host application program. The usual response would clear the screen. However, the host application may redisplay the screen contents or may display a new screen.

Enter

Enter is used to signal the end of each normal command or response. The IBM 7171 passes the converted ENTER sequence to the host application program for processing.

Test Request

When Test Request sequence is sent, the sequence is converted to the 3270 data stream command and sent to the host application program for processing.

Cursor Select

The Cursor Select sequence tells the IBM 7171 that the selector-pen- detection function was performed on the keyboard. This sequence is converted to the 3270 data stream, and together with the cursor x/y position, are sent to the host application program for processing.

Program Attention Keys

When any one of the three PA key sequences is transmitted, the IBM7171 forms the 3270 data stream including the AI byte. This data stream is then sent to the host application program.

Program Function Keys

When any one of the 36 PF key sequences is transmitted, the 3270 data stream including the AI byte is formed and sent to the host application program.

Other Key Commands

Redisplay
Receipt of the Redisplay sequence causes the display to be cleared and the data to be redisplayed.

Local Print
Receipt of the Local Print sequence causes an image of the screen to be printed to a hardcopy device connected to the auxiliary printer port of the terminal.

DUPlicate
The DUP sequence sends a command to the terminal that puts the DUP character on the screen and repositions the cursor to the beginning of the input area. When this screen is eventually sent to the host application program, the DUP character signals to the application program that a 'duplicate' operation is intended for the rest of the field in which it is located.

Field Mark
The Field Mark sequence on the terminal causes a special character to be placed in the screen buffer. When this screen is eventually transferred to the host application program, the special character in the buffer indicates the location of the end of a field on an unformatted screen or the end of a subfield in a formatted buffer.

Defining a Reset Keyboard Character Sequence: There are 7 special keyboard functions which are handled immediately by the IBM 7171 as they are received. They are exempt from the "keyboard lock" pacing provided for other keyboard input. For a detailed description of these reset keyboard functions refer to "Chapter 3. Using ASCII Terminals on the IBM 7171" on page 3-1. The sequences are terminal type dependent. These sequences are stored in a string which consists of a series of 8 characters followed by a terminating X'FF'. The data is positional, and the IBM 7171 expects each one to be identified. The order of the bytes is as follows:

1. Reset introducer (X'FF' = NONE IF NOT USED)
2. Master Reset
3. Character Error Reset
4. Keyboard Unlock ("3270 RESET")
5. Type-ahead Purge
6. Pacing Start
7. Pacing Stop
8. Operator console Toggle.

The pacing control must be provided by single control characters, and on all common devices this function is provided by DC1 (XON) and DC3 (XOFF). The other functions can be invoked by either single control characters, or by two-character sequences, beginning with a common introducer key.

This optional **Reset Introducer** can be coded as the first parameter in the Reset Keyboard Character Sequence. Suppose the user wants to assign some of the reset functions to keys on the terminal which generate ESC sequences. Define the first RCHRS character to be ESC, after which the other entries in the remainder of the string (except for "Pacing Start" and "Pacing Stop") define the second character of a two-character ESC sequence. The introducer need not be exclusively used in the RCHR strings. When the special introducer character is detected, a flag is set. If the next character is found in the remainder of the RCHR string (except "Pacing Start/Stop"), then the special function is invoked. If the second character is not matched in this list, then both the introducer and the second character are placed into the input buffer where they will be processed at normal priority following the usual keyboard lock protocol. Thus having ESC X as master reset does not preclude the definition of ESC A as cursor movement.

The character at position 8 of the RCHRS string is used as the Maintenance facility toggle. See "Chapter 9. Special Maintenance Facility and System Messages" on page 9-1 for complete details.

If the terminal uses the default reset keyboard character sequence, the following 7 ASCII control characters are reserved for these special functions:

X'07' Master Reset
X'12' Character Error Reset
X'14' Keyboard Unlock (3270 Reset)
X'18' Type-ahead Purge
X'11' Pacing Start (XON)
X'13' Pacing Stop (XOFF)
X'17' Operator console Toggle

Keyboard Assignment Considerations

There are over 60 unique keyboard functions supported. Since most terminals do not have 60 function keys, special techniques must be used to define any additional keys. One approach is to use a "shift" key (it is held down while pressing another key). Such a key is the CTRL key (marked ALT on the IBM 3101), which typically permits the generation of all 33 ASCII control characters from the keyboard. On the IBM 3101 with the ALT key depressed and pressing graphic character "A" produces the ASCII control character SOH. Another approach is to use a special "introducer" function key which sends a two-character sequence. In the case of the IBM 3101, eight Program Function(PF) keys provide such a capability. Finally, some terminals have a special function-shift key which embeds the character generated by the next normal graphic key pressed in the middle of a special three-character sequence (e.g. SOH-char-CR is sent).

The first task is to segregate the useable keys. Some keys have alternate functions (shifted and unshifted for example). It is important to distinguish the codes which can easily be generated (unshifted) from those which required two fingers. For example, on the IBM 3101, the best keys are Backspace, Return, and Tab (because they are large and are closest to the center of the keyboard). Backtab and SEND would be very good, but they do not transmit anything. The next best keys are the cursor movement keys and DEL (they are on the right side and most people are right-handed). Then come ERASE EOF, ERASE EOS, and ESC. Finally, we have the ALT keys (which require that one hand be used to hold down ALT while the other presses the key) Home, CLEAR, ERASE INPUT, and the PF keys. All other keys on the IBM 3101 do not transmit. Some terminals have a pad of twelve or more keys arranged in a 3x4 layout similar to a real 3270. Often it is tempting to use this pad for 12 PFKs, no matter how they are named, because much of the IBM documentation and assignment of meaning to PFKs is based on a particular layout of the PFK pad.

The next step is to evaluate the relative importance of the various functions available in the application environment under consideration. Usually ENTER, cursor movement, Erase EOF, Newline, Tab, DEL, INSERT, and the first 12 PFKs are most important. Sometimes CLEAR is critical (as in CMS). Often ERASE INPUT is a dangerous key to have. In the distributed tables there is no ERASE INPUT key assignment for most terminal types. Specific applications may have a need for CUR SEL or TEST REQ while most systems do not use them. The user must balance the keys available on a specific terminal with the needs of the applications.

It is usually a good idea to select keyboard assignments based on the name of the key itself, however key location can be important. If "ERASE EOL" is written on a key it may be a good idea to make it emulate the ERASE EOF function because this simplifies the documentation problem and reduces the amount a user must remember. These are not the only considerations. Also to be considered are key size, right-handedness, the proximity of "bad" keys (like a Master Microcode Reset key), and typamatic behavior to consider. Typamatic is critical for cursor movement, useful for Tab, and useless for ERASE EOF.

Keyboard extension techniques depend on the manufacturer's assignment of control sequences to the supplied edit function keys. On the IBM 3101, where all function keys transmit sequences beginning with ESC, the ESC key itself is ambiguous and should therefore be avoided as a manual sequence introducer. Therefore, the ERASE EOS key is recommended as the first key of two-key sequences. Other terminals assign control characters to the edit function keys (BS for Cursor Left, LF for Cursor Down, etc.). On such terminals the use of ESC is recommended as a function introducer key.

Once the function introducer is chosen, there are some recommended guidelines for selecting the second key of the two-stroke sequence for keyboard extension. One consideration is placement on the keyboard. The top row of the keyboard is commonly used, in conjunction with the function introducer, to represent the first 12 PFKs. This was chosen because some models of real 3270 terminals have PFKs in this location. It is also useful to assign the second twelve PFK values to the task of function introducer followed by the second row of the keyboard (QWERTYUIOP). But this is not always possible since these characters may be used in escape sequences generated by other keys on the terminal. For example, some terminals have a BACKTAB key which sends ESC-I. If ESC is the function introducer, the user must choose between using the BACKTAB key for its natural meaning and using ESC-I as PFK20. In some environments it may be possible to take advantage of the case (ESC-I is BACKTAB, ESC-i is PFK20), but this would be risky.

When followed by a keyboard function key, the function introducer is sometimes used to mean "more" (DEL means Delete Character, Function-DEL means Erase EOF), or "opposite" (Function-Tab means Backtab), or "looks like" (Function-Semicolon is Field Mark because FM prints as a semicolon on most 3270 models). But, in the final analysis many choices are a matter of personal preference.

4.3.4 Terminal Output

CSS Pointers and Strings

Each terminal must perform certain hardware functions upon receiving a character sequence. Normally we think these functions occur when the associated key is pressed on the keyboard. However, on a full duplex terminal, the function is performed only after a string has been transmitted from the keyboard to the IBM 7171, and another string has been sent back to the terminal.

The IBM 7171 is not required to send back the same string that the keyboard transmitted. For example, it is convenient to simply associate keys with the functions printed on them. Thus if there already is a key marked HOME it is natural to use it to emulate the 3270 HOME function. In a full-screen application, however, the 3270 HOME key requests that the cursor be positioned at the first unprotected position on the screen. When the user presses the HOME key on an IBM 3101 (an example of an ASCII terminal) a two-character sequence "ESC-H" is sent to the IBM 7171. If the system simply echoed back these two characters, the cursor would be positioned by the IBM 3101 at the upper left corner of the screen. Instead, a four character sequence "ESC-Y-row-col" which positions the cursor at the screen address associated with the first logically unprotected character position is sent. This mapping of input string to different output strings is performed by IBM 7171.

These output strings are called **Control Sequence Strings (CSS)**. They are specific to each terminal and must be found in the terminal manufacturer's specification manual. There are 19 defined Control Sequence Strings, but CSSs 7, 8, and 19 are not used.

Following is a list of each of the CSSs, and their meanings:

1. **Reposition** - This string is required to generate the terminal-specific control character sequence to position the cursor. When it is received by the terminal, the cursor is positioned according to the data sent in the string. As such some characters or character strings in this control string are variable and must be dynamically generated from the current cursor position. The dynamic generation of cursor position is performed by the routines shown in the following table:

```
X'FE'  BINX      Binary column value
X'FC'  BINY      Binary row value
X'FA'  CHARX     Character column value
X'F8'  CHARY     Character row value
X'F6'  USERX     User generated column value
X'F4'  USERY     User generated row value
X'F0'  HARDCOPY  Hardcopy terminal positioning
```

Figure 4-2. Cursor Positioning Routine Names

   a. The two routines BINX and BINY each form a binary position value of one byte length by adding the ORIGIN value to the column or row number. This is the most common cursor positioning method, and since the origin value is usually the ASCII "blank" (X'20'), the resulting values are from X'20' to X'37' for the rows and from X'20' to X'6F' for the columns of a terminal with a standard 24x80 character display. If ORIGIN=0 is specified, the BINX and BINY routines generate the values X'00' through X'4F' to address 80 columns. Origin=1 causes the values X'01' through X'50' to be generated in this case.

	Origin not coded			Origin=0		Origin=1	
row/column	HEX	ANSI	Char	HEX	ANSI	HEX	ANSI
0	20	2/0		00	0/0	01	0/1
1	21	2/1	!	01	0/1	02	0/2
2	22	2/2	"	02	0/2	03	0/3
	...						
16	30	3/0	0	10	1/0	11	1/1
	...						
23	37	3/7	7	17	1/7	18	1/8
	...						
33	41	4/1	A	21	2/1	22	2/2
	...						
79	6F	6/15	o	4F	4/15	50	5/0

b. The ANSI X3.64 standard for terminal control sequences defines instead the use of character text strings for numeric values. In this scheme, column 20 would be represented in the proper control string context by the ASCII character string "20" (X'3230'). The two routines CHARX and CHARY will produce such numeric character parameters for terminals that conform to this convention. When this form of addressing is used, the ORIGIN parameter should be specified as the number (not the character) 0 or 1 (for example ORIGIN=0 or ORIGIN=1 as for binary position values) depending on the numbering convention used on the target terminal for the first row or column, since the value "blank" (X'20') does not make sense in this notation.

	Origin=0			Origin=1		
row/column	HEX	ANSI	CHAR	HEX	ANSI	CHAR
0	30	3/0	0	31	3/1	1
1	31	3/1	1	32	3/2	2
2	32	3/2	2	33	3/3	3
	...					
16	3136	3/1 3/6	16	3137	3/1 3/7	17
	...					
23	3233	3/2 3/3	23	3234	3/2 3/4	24
	...					
33	3333	3/3 3/3	33	3334	3/3 3/4	34
	...					
79	3739	3/7 3/9	79	3830	3/8 3/0	80

c. USERX and USERY invoke user supplied cursor positioning routines. The user may add a new cursor positioning algorithm to support a non-standard terminal. A new Terminal Definition Table must be created in NV-RAM with a modified cursor positioning CSS string. When the cursor positioning control string contains X'F6', the IBM 7171 will call a user supplied cursor x-positioning routine located in NV-RAM. The column value (0-origin, i.e. the first column = 0000) is passed to a special USERX routine via the AX register. Similarly, if the control string contains X'F4', a call to a user supplied cursor y-positioning routine occurs and the row value (0-origin, i.e. the first line = 0000) is passed to a USERY routine via the AX register. The user routines must return the converted x or y data to be output to the terminal in the AX and DX registers. On return the user must load the CX register with a count of the number of bytes (0 to 4) to be sent to the terminal and issue a return instruction X'CB'. The data will be sent to the terminal in the order - AL, AH, DL, DH. For example, to position the cursor to the third line on a display, the IBM 7171 loads X'0002' into the AX register and calls the user supplied y-positioning routine. If the user's terminal required a X'22' to perform this movement, his routine would return X'22' in AL and a byte count of X'0001' in CX. In this case data returned in registers AH, DL, and DH are ignored. All registers are saved before the call to the user routine and restored after the return.

The user may enter his routines using the "Store to CPU Board Memory" command of the Maintenance Facility. The routines must be entered as executable Intel 80186 object code. The interface to these customized cursor positioning routines is defined below.

segment:offset	contents (byte reversed)	description
DC00:02F6	9E6C	USERX offset address
DC00:02F8	00DC	USERX segment address
DC00:02FA	9E6C	USERY offset address
DC00:02FC	00DC	USERY segment address

The user must modify the offset addresses to point to his routines which must be located in an unused part of NV-RAM. The segment addresses do not need to be changed. The offset addresses as delivered point to a dummy routine in ROM at DC00:6C9E which sets the byte count in register CX to zero and does a return. For example, to add a USERX routine at DC00:400 and a USERY routine at DC00:420 enter the following Store commands using the Maintenance Facility:

```
s DC00:2F6 0004
s DC00:2FA 2004
s DC00:400   object code for USERX, X'CB'
s DC00:420   object code for USERY, X'CB'
```

d. HARDCOPY invokes a routine which attempts to provide a useful approximation to cursor positioning on a hardcopy terminal, which is assumed to respond in the usual way to ASCII CR and LF. Cursor left and right are both used as defined in CSS strings 5 and 6 and any change of row causes movement down the page.

2. **Erase EOL** - This string is the terminal defined control sequence to generate the standard 3270-type ERASE EOL (erase to end-of-line) function.

3. **Local Print** - This string is sent to terminals which can print an image of the screen to a hardcopy device connected to an auxiliary printer port of the terminal if it exists. It will invoke the local print function on this terminal.

4. **Tone** is the control sequence to generate an audible alarm ("beep").

5. **Cursor Left** is the control sequence string to move the cursor on the screen one character position to the left.

6. **Cursor Right** is the control sequence string to move the cursor on the screen one character position to the right.

7. -- not used --

8. -- not used --

9. **Signal Insert Mode** is a string which is sent to the terminal when the user has entered insert mode. The 3270 terminal has an insert mode indicator. Transmitting the terminal-defined "Enter insert mode" string to an ASCII terminal should generate an equivalent indication for the "insert status" on that terminal. Some terminals have one or more lights or indicators which can be turned on or off by program control. On other terminals, the cursor might be set to blink when in insert mode but not to blink normally.

10. **Signal End Insert Mode** is a string which should turn off the insert mode indicator when the terminal user leaves insert mode.

11. **Disconnect** - This is a string sent to the terminal on a host initiated or keyboard initiated line drop.

12. **Clear** is the control sequence to execute the 3270-type CLEAR SCREEN function.

13. **Terminal Initialize** is a string which is sent to the terminal at initial connection (dial up, or power on) after the terminal type has been determined. This string may be nothing more than a "clear screen" request, or it may reset internal terminal logical states. For example, on a terminal conforming to ANSI X3.64 it might be appropriate to send the RIS (Reset to Initial State) control function to clear any inappropriate protected fields, partial screens, insert or other modes, and other logical states left over from a previous session with another computer system.

14. **Illegal ASCII Char** is a character or string which is written to the terminal to represent an unprintable character. The TDTs supplied with the distributed system use a colon (:) for this purpose. However, on certain terminals it may be desirable to choose a less ambiguous character. For instance, if the user has only Model 2x versions of the IBM 3101 terminal, a suitable alternate choice might be:

    ```
    ESC,E  (X'1B45')
    ```

    which writes an unusual graphic character (normally called the "SEND MARK") which would produce a unique display. On the other hand, on the IBM 3101 Model 1x it is possible to display a square blot on the screen with the more complicated

    ```
    DLE,STX,DEL,DLE,ETX  (X'10027F1003')
    ```

    Although this string contains five control characters, it produces the required result to be the illegal character token: it displays something unusual in the screen position, occupies only one space on the screen, has no effect on the characters which follow or precede it, and when done it has advanced the cursor one position to the right. Screen wrap at end of line is also handled properly.

15. **Illegal APL Char** - This is a special string for displaying illegal characters in the APL mode. The normal setting for this string is:

```
n,BS,z    (X'6E087A')
```

The lower case here is misleading. The control sequence strings are never translated by the APL mode setting, so lowercase prints as uppercase. "N" overstruck with "Z" produces a fairly solid blot on most APL terminals.

16. **APL ON:** This provides a string which may exist on terminals to turn APL mode on to permit display of both APL and normal ASCII characters on the same screen image.

17. **APL OFF** - This string turns APL mode off again.

    APL ON/OFF provides two strings which may exist on terminals to turn APL mode on and off dynamically to permit display of both APL and normal ASCII characters on the same screen image. On most terminals with dynamic APL capability, the control character SO (Shift Out) turns APL mode on and SI (Shift In) turns it off. However, it is possible that some other terminals would use another control sequence for this purpose.

18. **Display Mode:** The IBM 7171 uses this control sequence string to support highlighting and basic color (see "Highlighting" for details).

19. -- not used --

Highlighting

SGR stands for **Set Graphic Rendition** and is borrowed from the ANSI X3.64 standard document where it is the name given to the control string which performs the highlight/color function. For this string to be invoked, the X'0800' FLAGS bit must be set ON in terminal header of the given terminal type (see "Terminal Header Information" on page 4-30). The SGR string in conjunction with the 18th Control Sequence String specifies the display modes **highlighting** and/or **basic color**.

Special routines (BINFLD and CHARFLD) are called from the 18th CSS string. Each routine operates in conjunction with the **SGR** string for this terminal type and the attributes of the current field, to perform the highlight/color function.

On an IBM 3277 or 3278 the only form of alternate display available is **highlighting**, and therefore only the highlight attribute is important. On the 3279, however, there is a mode of operation called **basic color** in which four different colors are assigned to the unprotected unhighlighted, unprotected highlighted, protected unhighlighted, and protected highlighted fields. The IBM 7171 can support this philosophy of display, but requires more flexibility than the 3279 hardware provides.

For one thing, while every color CRT has 6 colors plus black and white, there are substantial differences between the brightness of different choices. On most screens, blue is too dark to be read against a black background. Red is better, but is still uncomfortable to read. Yellow may be difficult to distinguish from green, and white is difficult to distinguish from cyan. Thus, selecting a usable set of four distinguishable colors can result in different choices for different devices. Furthermore, there are alternate modes of display, such as reverse video, blink, or underscore, which may be used even on a monochrome terminal to achieve four distinguishable field display modes. Finally, different applications may require different choices of the four display options.

To accommodate these requirements, each terminal type is associated with an SGR string which supplies a **primary** and an **alternate graphic rendition** of the four possible combinations of protect and highlight field attributes. A fifth parameter is supplied for non-display fields.

The field attribute of the current display field is used to index into the first (primary) or second (alternate) half of the SGR string to find the parameter which sets the terminal to the right mode. The index values are:

```
0 Nondisplay
1 Normal intensity unprotected
2 High intensity unprotected
3 Normal intensity protected
4 High intensity protected
```

The selection of the first or second half of the SGR string is made by the setup functions "Primary/alternate display of attributes" in "3.1.5 Setup Functions" on page 3-3.

The value of the byte selected in the SGR string can be used in the 18th CSS in one of two ways. If BINFLD (X'EE') is specified in the string, the contents of the byte are substituted into that position in the string directly as an ASCII character value. If CHARFLD (X'EC') is specified, the byte selected is assumed to contain two four bit fields representing a packed integer which should be turned into its ASCII character representation and substituted into the current position.

For example, the ADM31 family of terminals (including the TVI950) has a normal intensity mode which is triggered by ESC followed by left parenthesis (X'28') and a half intensity mode which is triggered by ESC followed by right parenthesis (X'29'). Unfortunately, the half intensity is usually too dim to be acceptable in a normally lighted room. Even so, half intensity may, in some cases, serve as an alternative. Thus, the SGR string (5 bytes of primary followed by 5 bytes of alternate) is:

```
X'2828282828',X'2829282928'
```

and the corresponding 18th control sequence string is:

```
ESC,(BINFLD) or X'1BEE'
```

In primary display mode, a X'28' will always be substituted after ESC in the output string, and all fields will be written in normal intensity. In alternate display mode, an X'29' will follow escape in normal intensity fields (whether protected or not) and they will be displayed in dim half intensity.

An increasing number of terminals conform to the new ANSI X3.64 standard, and many color terminals support the ISO extensions to this standard. There is a philosophy in this standard that parameters should be expressed as the ASCII character form of numbers embedded between an introducer of ESC, LBRACK and a alphabetic terminator. For display attributes, the terminator is lowercase "m." Under these standards the parameter for a normal field would be "0" and a highlighted field would be "1." Under the ISO extensions, the colors are standardized as two digit parameters in the range "30" to "37."

The SGR string for such a standard terminal which has a monochrome primary display mode and an IBM basic color selection as the alternate display would be:

```
X'0000010001, X'3737333631'
```

with a corresponding control sequence string of:

```
ESC,LBRACK,(CHARFLD),LCM or X'1B5BEC6D'
```

The CHARFLD routine will substitute the ASCII characters "0" or "1" in normal display, and the two digit character strings "37," "33," "36," or "31" in alternate mode.

4.3.5 Translate Tables

Different terminal types may be assigned to different host and terminal character translation tables. The supplied default tables are suitable for normal ASCII-EBCDIC and APL support. However, on devices with specialized character sets, or in countries with different national character assignments, the user may wish to construct his own specialized translate tables for specific terminals or applications.

The function of the host translate tables is to provide a means of changing character sets between the EBCDIC based host and the ASCII based representation of the terminal screen image in the IBM 7171. The internal screen image buffer contains an extended ASCII character set. For more information refer to "Appendix A. ASCII and EBCDIC Data Conversion Tables." Similarly, the function of the terminal translation tables is to provide a means of changing character sets between the extended ASCII code used in the representation of the terminal screen image and the ASCII based terminal.

4.3.6 Steps in providing User-Generated TDTs

In order to provide new IBM 7171 terminal types, technical information must be obtained for the terminal type to be generated. The required technical information is available in a manufacturer's technical reference manual.

The following is a suggested approach to "building" up a Terminal definition Table:

1. Define a minimal set of input sequences sufficient to logon AND logoff. At minimum include: Enter, PF1 ... , Left, Right, Up, Down, Home, Clear, Enter.

2. Define Reset Chars for Pacing only.

3. Define Output sequences for Cursor position, EEOL and clear. Do not set the Flag for highlighting.

4. Logon and test cursor positioning and EEOF. Modify those sequences until screen output is correct.

5. Add terminal initialize and highlighting output sequences. Turn on the highlight flag and define Graphic Rendition string. Test and modify those sequences until screen output is correct.

6. Add Insert on/off, APL, etc. Test and modify those sequences until screen output is correct.

If terminal screen output is correct, the input sequences will be technically easy to do. If the terminal screen output is NOT correct either the device cannot be defined to look like a 3270 terminal or technical assistance is required.

Often terminal keys do NOT transmit and produce undesired effects, e.g. CLEAR key clears screen and does not TRANSMIT a sequence to the Host. Such keys are UNUSABLE. If Cursor keys do not transmit, an alternate sequence MUST be defined. [Ctrl-R, L U D] is one solution.

Remember that Reset Characters have priority over input sequences. Reset character conflicts are probably the second most common "bug" in terminal definitions (the most common is wrong parity/baudrate in PORTS).

It is recommended that new terminal tables should be tested prior to putting a new table into operation.

4.4 Organization of NV-RAM

NV-RAM is divided into 4 basic areas:

1. Data area 1
2. Port configuration data
3. Data area 2
4. Terminal Definition Table (TDT) area.

The organization of NV-RAM is as follows:

```
DC00:0000   +-------------------------------------------------+
            | Data Area 1                                     |
            |                                                 |
DC00:0010   +-------------------------------------------------+
            | Ports Area                                      |
            |                                                 |
DC00:0210   +-------------------------------------------------+
            | Data Area 2                                     |
            |                                                 |
DC00:02A8   +-------------------------------------------------+
            | Terminal Type Messages        D                 |
            |                           T   E                 |
DC00:02EE   |                           E   F                 |
            | Reserved                  R   I   T             |
DC00:0300   |                           M   N   A   A         |
            | Terminal Names            I   I   B   R         |
            |  Information              N   T   L   E         |
DC00:       |                           A   I   E   A         |
            | TDT's                     L   O                 |
            |                               N                 |
            +-------------------------------------------------+
```

Figure 4-3. IBM 7171 NV-RAM layout

IMPORTANT - All of NV-RAM is located in memory on the CPU board starting at segment X'DC00'.

Following is a brief description of the data contained in each area:

Data area 1
This data area contains statistical information about NV-RAM.

Port Configuration Data
This area contains the information necessary to configure the 64 possible RS-232-C ports. This block of memory contains the following information for each port:

- Number of rows on screen
- Number of columns on screen
- Baud rate of device connected
- Flags for device connected
- Address of default terminal type.

Data Area 2

This data area contains error message information for the operator. Error messages are stored in NV-RAM so that they are not erased when the system is powered down.

TDT Area

This area contains the following information:

1. The 'ENTER TERMINAL TYPE:' message
2. The 'VALID TYPES ARE:' message
3. The complete list of terminal types (including those in ROM)
4. All user supplied TDTs.

4.4.1 Data Area 1 Layout

Following is a list of the data stored in Data Area 1 of NV-RAM: erase write optimization

Address	Data Meaning
0000	Length of NV-RAM. Defaulted to X'2000' (8K bytes).
0002	Address of the first unused location in NV-RAM.
0004	Number of bytes to be allocated to the 3270 data stream buffer. Defaulted to X'27C0'.
0006	Erase/Write optimization buffer size. Zero implies no optimization. Defaulted to X'09F0'.
0008	System Control Flags. Defaulted to 0.

Flag Bit	Meaning
40	In byte 1 of the word. On or 1 = 3277 Emulation Off or 0 = 3278 Emulation

Address	Data Meaning
000A	Address of logoff message which is sent in the predefined message to host disconnect method.
000C	Sense information for synchronous status.
000D	Status information for synchronous status.
000E	Sense information for asynchronous method of host disconnect. See note below for definition of sense byte.
000F	Status information for asynchronous method of host disconnect. See note below for definition of status byte.

Sense Byte for Line Drop (Default value is '00')

Sense Flag Bits	**Meaning**
80	Command Reject
40	Intervention Required
20	Bus Out Check
10	Equipment Check
08	Data Check
04	Unit Specify
02	Control Check
01	Operation Check

Status for Line Drop (Default value is X'04')

Status Flag Bits	**Meaning**
80	Attention
40	Status Modifier
20	Control-Unit End
10	Busy
08	Channel End
04	Device End
02	Unit Check
01	Unit Exception

4.4.2 Ports Area Layout

Each of the 64 possible ports is configured by 8 bytes. Each port is addressed **sequentially** from address X'DC00:0010'. The first 8 bytes are dedicated to port 0, the final 8 bytes are dedicated to port 63. Following is the definition of the 8 configuring bytes:

Byte # **Meaning**

0 Number of rows on screen. If left 0, 24 rows is the default value.

1 Number of columns on screen. If left 0, 80 columns is the default.

2-3 Baud Rate. Stored as a hexadecimal number, byte-reversed. For example, a 9600 baud terminal would have its baud rate store as X'2580'. However, since there is byte reversal, byte 2 would be X'80' and byte 3 would be X'25'. If the baud rate is null (0000), or is invalid, the terminal will be in autobaud detect mode. When a terminal is in autobaud detect, a user must press the carriage-return key. The IBM 7171 will then automatically determine what baud rate to use. The following baud rates are autobaud detectable: 300, 600, 1200, 1800, 2400, 3600, 4800, 9600 and 19200 baud.

The following baud rates are supported, but are NOT autobaud detectable: 50, 75, 110, 134, 150, 2000, and 7200 baud.

4-5 Terminal Flags.

Byte 4 Flags -

Bits	**Meaning**
20	Stick bit. Off or 0 = Parity bit is interpreted as even or odd, On or 1 = Parity bit is interpreted as mark or space.
10	Parity. Off or 0 = Odd parity (or if Stick bit is on always a 1 - Mark parity), On or 1 = Even parity (or Space parity if Stick bit is on).
08	Parity Enable. Off or 0 = Parity Disabled, On or 1 = Parity Enabled
04	Stop Bits. Off or 0 = 1 stop bit, On or 1 = 2 stop bits
02-01	Data Bits. 00 = 5 data bits, 01 = 6 data bits, 10 = 7 data bits, 11 = 8 data bits.

Byte 5 Flags -

Bits	**Meaning**
02-01	Type of connection. 00 = Let 7171 detect, 01 = Switched Network (Telephone Line), 10 = Leased Line, 11 = Direct Connect If both bits are zero, the 7171 will automatically determine the type of line that is attached.

6-7 Terminal Header Pointer. This can be one of two things: An address of or an index to a Terminal Header. The specified terminal will then be automatically selected at terminal power-up. If this value is left zeroed, the user will be prompted with the **'ENTER TERMINAL TYPE:'** message.

Addresses are offsets relative to segment X'DC00', and are stored byte-reversed. Addresses can point to either NV-RAM or ROM.

Table addressing in the 7171 frequently uses pointers. In this context the term "pointer" has a unique definition:

```
If bit 15 (the MSB) = 0

THEN
  The contents of the pointer are a relative
  (to segment X'DC00') address (max of 32k).
ELSE
  The contents of the pointer are an index that
   is converted into a displacement into the
   beginning of IBM 7171 ROM (relative address X'4000').
  Bits 6-0 of byte 6 are the ROM index and allow
   addressing  of the 128 entries (256 bytes).
  Thus pointer (index) X'8000' will address the
   ROM location the address of which is at X'4000'.

   See Figure 4-9 on page 4-47 for indices and corresponding
   ROM addresses.
   The location in ROM contains the relative address of
   the named table or character string.
```

Following is a list of possible indices defined (note that these are not addresses of terminals, but are rather indices into a list of terminal addresses).

Index	Terminal Type
8000	IBM3101
8001	Undefined
8002	TVI912
8003	TVI920
8004	TVI950
8005	TVI950R
8006	ADM31
8007	ADM3A
8008	VT100
8009	DM1520
800A	DM1521
800B	DM3045
800C	TYPETERM
800D	HARDCOPY

These values are also byte-reversed in storage.

As an example, consider the following 8 bytes:

```
00 84 00 4B 1A 03 02 41
```

This port is configured for 24 rows, 132 columns, a baud rate of 19200, even parity, parity enabled, 1 stop bit, 7 data bits, direct connect, and will automatically be assigned the TDT at address DC00:4102. Here is another example:

```
18 50 00 00 0A 00 00 80
```

This port is configured for 24 rows, 80 columns, autobaud detect, odd parity, parity enabled, 1 stop bit, 7 data bits, default connection, and will automatically connect as an IBM3101.

4.4.3 Data Area 2 Layout

Following is a list of the data stored in Data Area 2 of NV-RAM:

Address	*Data Meaning*
0210	Address of first error message in queue.
0212	Address of last error message in queue
0214	Start of error message area

Each error message consists of 10 bytes. Error messages are stored sequentially, starting with message 0. Following is a description of the 10 bytes:

Byte	***Meaning***
0	Device Number
1	Message Number identifying error that occurred
2-3	Date of occurrence
4-9	Passed Parameters

Other words of consequence stored in this area are described below:

Address ***Data Meaning***

0290 Maximum time in milliseconds DSR can drop while the line is still considered to be connected. Default value is 100 milliseconds.

0292 Maximum time in milliseconds Receive Line Signal Detect (RLSD) can drop while the line is still considered to be connected. Default value is 500 milliseconds.

0294 Erase Write Optimization Baud Rate - Erase/Write Optimization can be enabled as a function of the baud rate. The word at DC00:294 specifies the baud rate above which no optimization will be performed. The default is 7200.

0296 Keyboard Unlock Delay

Not all IBM operating system software was designed with typeahead in mind. For example, in VM when an application program is invoked from the command line, there are a series of host write commands sent to update the screen before the application program gets control. These writes logically unlock the keyboard. If there is typeahead input when this occurs, that input will not be processed by the application, but by VM. To help minimize this effect, the IBM 7171 provides a programmable timed delay of the keyboard unlock by a host write command. This time delay is located at DC00:296. The word value at this location is a hex number representing the number of 32 millisecond time intervals between the time a keyboard unlock bit is received from the host and the time the keyboard is actually unlocked.

The default value is zero which means that there is no time delay. In most cases this will let typeahead input be processed by the program for which it was intended. For those cases where this is not true the system operator can chose a value for his installation which his users find satisfactory.

4.4.4 TDT Area Layout

The TDT area starts at address X'DC00:02A8', and is divided into the following parts (See Figure 4-3 on page 4-22):

Terminal Type Message Area
: This area contains the 'ENTER TERMINAL TYPE:' message, the 'VALID TYPES ARE:' message, and message area information.

Reserved
: This area must remain intact to maintain the integrity of NV-RAM

Terminal Names Information Area
: This area contains the list of valid terminals.

Terminal Definition Tables
: This area may contain the TDT's.

It is recommended that the reader examine a sample terminal table in IBM 7171 memory concurrently with reading this section. The IBM3101 terminal table will be used as an example in the sections dealing with the TDT's. The IBM3101 TDT is in ROM, but it follows the exact same form. The address of the IBM 3101 Terminal Header Information is found at address X'DC00:4000'. The layout of each of these divisions will now be given in detail.

Terminal Type Message Area

This area is broken up as follows:

Address	Definition of Stored Data
02A8	The length of the **'ENTER TERMINAL TYPE:'** message is stored here.
02AA	The **'ENTER TERMINAL TYPE:'** message is stored here. Any message may be put in its place.
02CC	The length of the **'VALID TYPES ARE:'** message is stored here.
02CE	The **'VALID TYPES ARE:'** message is stored starting at this location. Any message may be put here.

Extra space is left at the end of each of the messages. This is so the message can be changed without changing all of NV-RAM

Reserved Area

Warning: Do not alter this part of NV-RAM. This area (X'02EE' through X'02F3') is used by the IBM 7171 to determine the validity of NV-RAM. If the data expected is not found here, NV-RAM will be re-written with all default values. All user supplied information will be erased.

Address	Definition of Stored Data
02F4	The time in milliseconds to delay initilization of the terminal. Default value is zero.
02F6	The address (in this case both Segment and Offset) of USERX.
02FA	The address (in this case both Segment and Offset) of USERY.

Terminal Names Information

This is the information that appears immediately after the the 'VALID TYPES ARE:' message. It is the list of all valid terminal names that will be recognized.

This area provides the following information:

- Terminal Name
- Layout of Terminal Name message that immediately follows the **'VALID TYPES ARE:'** message
- Pointers to the Terminal's TDT.

The information is stored sequentially starting at address X'DC00:0300' in the following order:

1. Terminal Name, in (upper case) ASCII. The terminal name *must* be space-filled to 8 characters.
2. Pointer to TDT, 2 bytes, byte-reversed. The pointer can take 2 forms:
   a. An address of a TDT.
   b. An indirect pointer: an index into ROM. These are the same values as those used in ports configuration. For a list of these values, see "4.4.2 Ports Area Layout" on page 4-24.
3. An optional Carriage control word, or list transfer word. This word can take the following forms:
   a. X'FFFF' indicates that there is to be a carriage return after this terminal name is printed on the screen.
   b. X'FFFE' indicates that all terminal names following this are to be 'hidden'; that is, they will not be printed on the screen.
   c. If the high bit is set, and the first byte is not X'FF', the remaining 15 bits, are taken to be an address to transfer to. That is, the next terminal name will be sought at the new address.
   d. If the word is null, X'0000', then the end of the list is indicated.

   Please note, that the word X'0000' *must* appear someplace. If it does not appear, the IBM 7171 will not find the end of the list, and results are unpredictable.

The information for each valid terminal follows sequentially after the previous one. Thus, this area contains a continuous list of terminal information.

Terminal Definition Tables.

The user defined TDTs appear in NV-RAM following the Terminal Names Data, just described. All pointers to TDTs actually point to the first byte of the Terminal Header Information, which happens to be a status flag byte.

Note: Since all of the TDTs use many different levels of indirection, it is possible for several TDTs to share data. For instance, if it is desired to have all of the terminals use the same 'Reset Characters' string (discussed in detail later), there need be only one string of 'Reset Characters', and each TDT can point to it. In this way, it is possible to optimize the amount of space used.

The TDT is broken up into the following three parts:

1. Terminal characteristic data and basic pointers (Header information)

2. CSSs and pointers

3. Input Parse Tables.

To better understand this part of the document, the reader might want to examine IBM 7171 memory using the Maintenance Facility (described in Chapter 9) and follow each of the examples listed.

Terminal Header Information: The first part of the TDT is the Terminal Header Information. The Terminal Header has 9 items and each item occupies two bytes of storage. These 18 bytes provide basic information about the specific terminal.

Item	Definition of Stored Data
1	Status Flags
2	Pointer to Initial Parse Table Entry
3	Pointer to First CSS Pointer
4	Cursor Base
5	Pointer to Host Translation Table Addresses
6	Pointer to Terminal Translate Table
7	Pointer to Reset Character String
8	Delay
9	Pointer to Set Graphics Rendition String

A description of each of these fields follows:

1. **Flags.**

The Flags parameter can be used to set bit flags which describe special terminal characteristics. The following flags can be specified:

Byte 0 Flags -

Bits	**Meaning**
X'80'	Hardcopy terminal
X'40'	Can switch from APL to non-APL and back under program control (usually SI and SO characters)
X'10'	Set = bit paired terminal (used only in APL support). Off or 0 = type paired terminal
X'08'	Highlighting can be performed
X'04'	Highlighting is a mode of output, where characters written after an escape sequence are displayed highlighted, rather than an attribute byte, where characters in the following screen positions change display without themselves being rewritten.

Byte 1 Flags -

Bits	***Meaning***
X'40'	Can display character in the lower right corner without scrolling the screen.
X'20'	Use "real" 3270 algorithm to suppress all nulls on READ MODIFIED response. If not specified embedded nulls will be returned to the program as blanks.
X'10'	On or 1 = default pacing to off, Off or 0 = default pacing to on.

2. **Pointer to Initial Parse Table Entry.**

   This points to the keyboard parse table data. The parse table parses the input string from the keyboard and determines an action to take. The Input Parse Table is discussed in "Input Parse Table."

3. **Pointer to First CSS Pointer.**

   This points to a list of pointers. The list, in turn, points to the individual CSSs (control sequence strings) described in "CSS Pointers and Strings" on page 4-14.

4. **Cursor Base.**

   The cursor base allows a numeric or character base to be used in generating direct cursor addressing strings to be specified. Some terminals number the rows and columns starting at 0, while others start at 1. For some terminals, a single ASCII character represents a row or column number based on its binary value in the ASCII character set. Other terminals convert the row or column number to a "printable" numeric character string. The cursor base is used in any of these cases to specify the proper value for the first row or column. The cursor base is added to the row or column number addressed to generate the proper cursor locate sequence.

5. **Pointer to Host Translation Table Addresses.**

This is a pointer to a list of Host Translate Table addresses. The list of Host Translate Table addresses(offsets from segment X'DC00') must be 20 bytes long, and must appear in this order:

a. Host Read 3277 (ASCII to EBCDIC)
b. Host Read 3277 Attribute and APL
c. Host Write 3277 (EBCDIC to ASCII)
d. Host Write 3277 Attribute and APL.
e. Host Write 3278 (EBCDIC to ASCII)
f. Host Write 3278 Attribute
g. Host Write 3278 APL.
h. Host Read 3278 (EBCDIC to ASCII)
i. Host Read 3278 Attribute
j. Host Read 3278 APL.

The address of the standard list is found '40E2' (See "4.5.4 ROM Pointers to Internal Information" on page 4-47.) Refer to Appendix A for actual Translate Tables.

6. **Pointer to Terminal Translate Table.**

   This points directly to the start of the Terminal Translate Tables. The address of the default tables can be found at X'40E0' (See "4.5.4 ROM Pointers to Internal Information" on page 4-47). These tables translate keystrokes from the terminal. They are either left alone, or changed into flagged data. The following Translate Tables must appear sequentially in the following order, and must be of the following length:

   a. Normal ASCII Table (256 bytes) is used to convert a character from the IBM 7171 screen image buffer represented by internal extended ASCII code to normal ASCII code for output to the terminal.
   b. Normal ASCII Input Translate Table (128 bytes)
   c. Typewriter Paired APL Table (256 bytes) is used to convert a character from the IBM 7171 screen image buffer represented by internal extended ASCII code to typewriter paired ASCII code for output to the terminal.
   d. Typewriter Paired APL Input Translate Table (128 bytes).
   e. Bit Paired APL Table (256 bytes) is used to convert a character from the IBM 7171 screen image buffer represented by internal extended ASCII code to bit paired ASCII code for output to the terminal.
   f. Bit Paired APL Input Translate Table (128 bytes).
   g. Alternate Keyboard Input Translate Table (128 bytes).

   Note: For a description of the resident translate tables, see "Appendix A. ASCII and EBCDIC Data Conversion Tables."

7. **Pointer to Reset Character String.**

   The RCHRS parameter points to the start of a string of special reset keyboard characters which provide user control over error recovery and also define pacing protocol. The definition of this string is described in "Defining a Reset Keyboard Character Sequence" on page 4-11.

8. **Delay.**

   The Delay parameter makes it possible to specify the length, in milliseconds, of pauses between output strings being sent to the device. Delay causes temporary suspension of character output to the terminal for a specified time. It is used when the terminal has limited buffer capability and may require time to perform an operation.

9. **Pointer to Set Graphics Rendition String.**

   The SGR keyword points to the start of a string defining the variable parameters in the graphic rendition control string for the terminal (which sets highlighting, color, or reverse video). This string is described in "Highlighting" on page 4-18.

That is all of the positional data that is required. All of the necessary data is now identified via the above pointers. The CSS pointers and strings usually appear immediately following the header information.

Let us look at the first 18 bytes of the IBM3101 TDT as an example, It is found at address X'DC00:4000': In our example, the address at X'DC00:4000': is assumed to be X'52F2' The 18 bytes found at this address are:

```
00 00  4E 53  04 53
20 00  5E 42  72 4C
72 51  00 00  7B 51
```

This information tells us the following: The terminal flags are zeros. The first level input parse table is at address X'534E'. The list of CSS pointers is found at X'5304'. The cursor base is X'0020'. The list of host translate tables is found at X'425E'. The terminal translate table is found at X'4C72'. The reset character string is found at X'5172'. There is no delay, and the SGR string is found at X'517B'.

CSS Pointers and Strings: The third word of the terminal header information points to the first byte of the 19 CSSs. Each of the 19 entries is the address of the first byte of the given CSS. The addresses for all must exist, even if the string is not used. The addresses appear in this order:

1. Cursor Reposition String
2. Erase to the End Of Line
3. Local Print
4. Tone
5. Cursor Left
6. Cursor Right
7. --Unused
8. --Unused
9. Signal Insert Mode On
10. Signal Insert Mode Off
11. Disconnect
12. Clear
13. Terminal Initialize
14. Illegal ASCII Character
15. Illegal APL Character
16. APL Characters On

17. APL Characters Off
18. Display Mode
19. --Unused.

Note: All of the above addresses are byte-reversed in memory.

Each string starts at the address given in the above list of pointers and is terminated by the byte X'FF'. It is imperative that the string be terminated by X'FF'. If it isn't, the processor will think that the string is continuing on, and results may be disastrous. The bytes that appear in the CSS string are sent to the terminal exactly as they appear in memory, with the following exceptions:

X'FF' There will be some strings, such as those that are unsupported, which should be NULL. In these cases, the pointers should point directly to the CSS terminating byte, X'FF'.

X'FE' This is used in a Cursor Position String to indicate where a binary 'x' position, or column position goes. The column is added to the cursor origin, and sent instead of this byte.

X'FC' This is used in a Cursor Position String to indicate where a binary 'y' position, or row position goes. The row is added to the cursor origin, and sent instead of this byte. For example, if the row desired is 7, and the origin is X'20', the byte X'27' is sent to the terminal.

X'FA' This is used in a Cursor Position String to indicate where a character mode 'x' position, or column position goes. The cursor origin is added to the column number, and the ASCII string representing the hex byte is sent to the terminal.

X'F8' This is used in a Cursor Position String to indicate where a character mode 'y' position, or row position goes. The cursor origin is added to the row number, and an ASCII string representing the hex byte is sent to the terminal. For example, if the row desired is X'12', and the cursor origin is 1, then the two hex bytes, X'31' and X'33' (ASCII 1 and 3) are sent to the terminal.

X'F6' This is used in a Cursor Position String to indicate that a user supplied cursor x-positioning routine (USERX) will be called.

X'F4' This is used in a Cursor Position String to indicate that a user supplied cursor y-positioning routine (USERY) will be called.

X'F2' This indicates that a delay will occur equal to the number of milliseconds specified in the terminal header delay parameter.

X'F0' This is used in a Cursor Position String for a terminal type such as HARDCOPY or TYPETERM to indicate that cursor positioning should be approximated for a hardcopy device.

X'EE' This is used in a Display Mode String to select the routine used (BINFLD) to process the Set Graphic Rendition data for this terminal type.

X'EC' This is used in a Display Mode String to select the routine used (CHARFLD) to process the Set Graphic Rendition data for this terminal type.

For instance, if the characters "ESC" "[" row column were to be sent to the terminal for one CSS, the string might look like this:

```
1B 5B FC FE FF
```

If the command was to move the cursor to row 4 and column 5 and the cursor origin was X'20', the following string would be sent to the terminal:

```
1B 5B 24 25
```

The length of each Control Sequence String is limited to 16 characters with the exception of the initialization string (no. 13), which may be 256 bytes long.

Input Parse Table: Keyboard input processing is driven off a syntax table called the input parse table. The overall input parse table is divided into a main table and one or more next-character lookup sub-tables associated with each alternative possible in the left-to-right processing of the input control character string.

The main table will also be called the **first-level input parse table**. A sub-table that can be reached directly from the main table will be called a **second-level input parse table**; one that is reached from a second-level table will be called a **third-level input parse table**, and so on.

Each entry in parse table is a series of 4 bytes which is called a node. When a character is dequeued from the input typeahead buffer, the first byte of each node in the current parse table is searched until the first byte either matches the input character or matches one of four special bytes. When a character match is found, the 'action' field of the node (third and fourth bytes of the node) are scanned to determine what action to take. When a special symbolic match is made, the action is one of the following:

1. Signal invalid end with a beep at the terminal
2. Proceed with the table search using the byte to identify the search mode. This additional capability is provided to reduce parse size and is described in the following sections.

The nodes may be described using the following psuedo-code. The psuedo-code notation has the following form:

```
[label]  char[,param]  action target [(routine)]
```

label This term is optional. It is symbolic, and stands for a definition of the address at which the label appears.

char This term represents the ASCII characters which are being searched or one of four symbolic names which identify additional search mode activity. A list of ASCII character names and their hex values are found in Figure 4-6 on page 4-43. The four symbolic names and their hex values are:

END Stands for X'FF'

OTHERS Stands for X'FE'
INDEX Stands for X'F8'
MATCH Stands for X'FA'

The function of END and OTHERS is given here while the function of MATCH and INDEX are described later in this chapter.

END This signals the end of a subtable. When this character is found, the parse is invalid and a beep is generated at the user's terminal. Each sub-table must end with the END byte (X'FF'). The convention is to have the address field of an END node point to itself.

OTHERS This indicates that a different subtable should be searched. When this match is made, the target field of the node points to a sub-table which is searched immediately, using the same input character. OTHERS is a good tool for tying tables together. For example:

```
                    OTHERS     GOTO TYPPFK
                 . . .
TYPPFK              @1         CALL PFK1
                    @2         CALL PFK2
                 . . .
```

When two tables end in a long string of common matches, the first table can simply transfer control with OTHERS to the common ending table. Thus in this example, when OTHERS GOTO TYPPFK is encountered, the input scan leaves the current table and resumes the character matching process starting at the TYPPFK node.

The reader may wish to come back to this example after reading the rest of this section.

param This optional term is the parameter that is to be passed to the routine. Byte 1 of each node provides a parameter value to be stored for transmission to a routine. The parameter is stored in the Data Control Block for that device. If the parameter is 0, the current parameter is not changed. Whenever a non-zero parameter is found, it is stored in the Data Control Block for that device. The parameter is then used by the routine to determine a course of action. For example, the ATTENTION routine uses the parameter to determine which attention routine was initiated. For a complete list of possible parameters to pass to routines, see "4.5.3 Parameters to Pass to IBM 7171 Routines" on page 4-46.

action This indicates the action that is to be taken following a character match of byte 0 of the node. The action will be either:

- GOTO

  or

- CALL.

CALL indicates that a complete string was parsed correctly, and gives the routine to call. GOTO indicates that control is to be passed to another search list, and that the parse will continue.

If a CALL is specified byte 3 is NULL, X'00', and byte 2 provides the "routine" number, which is an index into an internal routine lookup table for the entry point of the routine requested. See Figure 4-7 on page 4-44 for a list of available routine numbers.

If a GOTO is specified byte 3 of the node is NOT NULL. Then bytes 2 and 3 form an address (remember byte-reversal) with a pointer to a next level input parse table, or a pointer to the special MATCH and INDEX strings.

target This is a symbolic way of specifying either the number of the routine to call, a pointer to the address of the next level parse table to search, or a pointer to an address of a MATCH or INDEX string.

The term "pointer" in this context has the same meaning as defined previously,that is

```
If bit 15 (the MSB) = 0

THEN
  The contents of the pointer are a relative
  (to segment X'DC00') address (max of 32k).
ELSE
  The contents of the pointer are an index that
   is converted into a displacement into the
   beginning of IBM 7171 ROM (relative address X'4000').
  Bits 6-0 of byte 2 are the ROM index and allow
   addressing  of the 128 entries (256 bytes).
  Thus pointer (index) X'8000' will address the
   ROM location the address of which is at X'4000'.

   See Figure 4-9 on page 4-47 for indices and corresponding
   ROM addresses.
   The location in ROM contains the relative address of
   the named table or character string.
```

routine This optional term is a comment to tell the reader what routine is indicated. It has no bearing on any byte values in the four byte entry.

Thus the pseudo-code represents four bytes of IBM 7171 storage which has the form:

Byte 0 Character or process indication byte (char)

Byte 1 Parameter Value (param)

Bytes 2-3 Address of next parse table or routine number to call (target)

Note: The input parse table does not parse the reset characters that are separately processed by the interrupt handler (Keyboard RESET, XON-XOFF, etc.), and which are described in "Defining a Reset Keyboard Character Sequence" on page 4-11.

When the user is typing in simple graphic characters (blank X'20' to tilde X'7E'), there is no need to refer to any table. The character is simply deposited in the internal screen image buffer at the current cursor position. When a control character is encountered, the first-level input parse table is examined to determine what action should be taken.

A sample first-level IBM3101 input parse table is:

```
IBM3101I     ESC    GOTO IBMESC           DC00:534E 1B 00 72 53
             HT     CALL TAB                        09 00 0A 00
             BS     CALL ENTER                      08 00 02 00
             CR     CALL NL                         0D 00 0B 00
             DEL    CALL DELETE                     7F 00 09 00
             SYN    CALL RESHOW                     16 00 0E 00
             VT,80  CALL DPFM        (dup char)     0B 80 13 00
             FF,7F  CALL DPFM        (fm char)      0C 7F 13 00
             END    GOTO *                          FF 00 70 53
```

Figure 4-4. First-Level Input Parse Table for the IBM3101 TDT.

In this description of the input parse tables each node is described in two ways:

1. The pseudo-code description.

2. A hex representation of the bytes indicated by the pseudo-code. Some nodes are prefixed by the address of the node.

All IBM 3101 control keys either transmit a single control character (HT,BS,CR,DEL,SYN,VT,FF), or a compound sequence beginning with ESC. The first entry in the table shown above identifies the ESC introducer. It specifies, using the GOTO parameter, that it is part of a control string which continues in the sub-table whose address is labelled IBMESC (see below).

The next 7 entries define single control character requests which in themselves constitute a control request. Local functions are invoked when a parse sub-table node matches a specific character and includes the CALL form. The valid routine names for CALL and the pertinent keyboard functions they invoke are specified in Figure 4-7 on page 4-44.

The last entry is an END value which is used to signal the end of the sub-table. Failure to include the END in a parse sub-table will permit the syntax scan to "fall through" into the characters of the next table that follows sequentially. This is incorrect.

Most of the other IBM 3101 control keys produce a two-character sequence beginning with the ESC control character. The parse sub-table for these keys is shown in Figure 4-5 on page 4-39.

```
IBMESC      J     GOTO IBMESC2       DC00:5372     4A 00 A6 53
            HT    CALL COLTAB                      09 00 1C 00
            CR    CALL COLBAK                      0D 00 1D 00
            A     CALL UP       (cursor up)        41 00 05 00
            B     CALL DOWN     (cursor down)      42 00 06 00
            C     CALL RIGHT    (cursor right)     43 00 04 00
            D     CALL LEFT     (cursor left)      44 00 03 00
            H     CALL HOME     (home)             48 00 0D 00
            I     CALL EEOF     (erase eof/eol)    49 00 0C 00
            K     CALL CURSEL   (erase input)      4B 00 11 00
            L,4   CALL ATTN     (clear)            4C 04 02 00
            W     CALL LPRT     (print)            57 00 08 00
            END   GOTO *                           FF 00 A4 53
```

Figure 4-5. Second-Level Input Parse Table for the IBM3101 TDT.

The Cursor Right key on the IBM 3101 sends the two-character sequence "ESC"-"C." When the "ESC" character is received the first-level parse table is scanned. The character matches the node in the table which specifies GOTO IBMESC. A GOTO node terminates the processing of the current character and stores the named label as the starting point for the scan to match the next character received. When the "C" is received, the "C" is matched to the CALL RIGHT node in the second level input parse table. This terminates the control string and calls the "Cursor Right" routine to perform the selected function.

If the sequence ESC,J,Z is encountered, the ESC would again cause the scanning to be continued in the second-level input parse table IBMESC. The J character will then be found in this table and will cause the third-level input parse table IBMESC2 to be entered, where scanning will continue for the character Z.

PARAM Field

The PARAM field is used only by the routines "ATTN," "SET," "RESET," "DPFM," and "DEFER." However, the parameter does not necessarily have to be specified on the final node. For example, suppose one also wished to use the eight "native" IBM 3101 Program Function Keys and identify them with PFK 1-8 on the 3270. This could be done by adding eight additional nodes to the second-level input parse table which is entered when ESC has been encountered as the first character of an input character sequence.

```
LCA,pf1   GOTO PEND
LCB,pf2   GOTO PEND
LCC,pf3   GOTO PEND
LCD,pf4   GOTO PEND
LCE,pf5   GOTO PEND
LCF,pf6   GOTO PEND
LCG,pf7   GOTO PEND
LCH,pf8   GOTO PEND
```

Then, if all four possible termination characters are to be supported, a final table in the following form would be needed:

```
PEND          CR        CALL,ATTN
              ETX       CALL,ATTN
              EOT       CALL,ATTN
              XOFF      CALL,ATTN
              END       GOTO,*
```

The first sub-table entries will cause the PARAM value of the appropriate pf-key to be stored in a control block, but the actual call to the ATTN routine will be deferred for one more character. Thus, the next to last character is the character which determines which pf-key was pressed.

Note: If the XOFF is used as a termination character than it can not be used as "pacing off" in the reset string.

DEFER

By calling the special DEFER routine, the same thing can be done for other (non-ATTN) keys. Assume, for example, it was decided to use the IBM 3101 PF8 key as a Backtab. The relevant entry in the second-level table would be:

```
LCH,X'0F'  GOTO DEND
```

The byte X'0F' is the routine number for Backtab (the desired routine), and is determined from Figure 4-7 on page 4-44. The final PFK terminator table would consist of entries in the form:

```
DEND          CR          CALL DEFER
        etc.
```

The DEFER routine looks up the X'0F' parameter in the control block and executes the requested function.

The Functions INDEX and MATCH

With only the previously documented NODE forms, terminal tables typically need long sequences of entries which are identical in form, changing only the matching character and parameter. For example, the 36 possible PFK values require 36 consecutive entries (and sometimes a second set of 36 values for the same keys shifted). To reduce the space which this occupies and simplify table generation, two functions called **INDEX** and **MATCH** may be used in byte 0 of the node statement.

INDEX and MATCH are functions which generate different parameter values given different inputs. The input to an INDEX or a MATCH is the next character in the input stream. Since the target field (bytes 2 and 3) of the node statement is used to identify the address of an INDEX or MATCH string, a second node is always coupled with the INDEX or MATCH and is used to determine the action and target field if a correct identification is found in the INDEX or MATCH functions. This second node takes the form:

```
     null,param  call routine
```

or

```
     null,param  goto sub-table
```

If a correct identification is not made within the MATCH or INDEX, then both nodes are passed over in the search list.

Index: The target field of the INDEX node points to a list of characters. The list is terminated by the byte X'FF'. The list of characters is searched for the input character. If the input character is found in the list, the zero-based index of the matched character relative to the beginning of the list is added to the parameter in the INDEX node, and the routine given in the coupled node is called with the new parameter. For instance, assume the input character was a X'31', and the following nodes were found in the search list:

```
INDEX     GOTO CHARLIST
NULL      CALL ATTN
```

Assume also that CHARLIST pointed to the following list of characters:

```
2F 30 31 32 FF
```

The character X'31' would match the X'31' in the target list, CHARLIST. The zero-based index of the match, 2, would be added to the parameter in the node. The resulting parameter, 2, would then be passed to the ATTN routine. If the input character is not found in the list, it is treated as a non-match, and the parse continues as usual, skipping both the INDEX and the NULL nodes.

Match: Match is very similar to index. The target string for match, however, is a series of pairs of values. The first character in the pair is the character to match; the second is a parameter value to be added. If the character is matched, instead of adding the index of the character that was matched, the value of the byte immediately following the matched character (the second byte of the pair) is added to the parameter. For instance, assume we had the same input character, X'31', and the following nodes were found in the search list:

```
MATCH,7   GOTO MATCHLIST
NULL      CALL ATTN
```

Assume also that MATCHLIST pointed to the following list of characters:

```
2F 31 30 05 31 06 32 07 FF
```

Every other character is examined for a match. In this case, the fifth byte is matched. The byte immediately following it, X'06' is added to the parameter found in the MATCH statement, 7, and the result is passed on to the routine in the next node statement. In this case, ATTN would be called with a parameter of X'0C'.

Many different terminals have quite similar requirements. First, some tables represent the same terminal with minor differences in keyboard assignments (which key is ENTER and which is Newline). Some tables are based on modest differences between several models of the terminal, or among options on the same model or microcode levels. Many terminals emulate other brands of terminals, but with improvements.

Using the above functions (INDEX, MATCH, DEFER, and OTHERS), one could create very condensed TDTs that could function for several closely similar terminals.

Reset Keyboard Character Sequence: There are 7 special keyboard functions which are handled immediately. They are exempt from the "keyboard lock" pacing provided for other keyboard input. For a detailed description of these reset keyboard functions refer to "Chapter 3. Using ASCII Terminals on the IBM 7171" on page 3-1. The sequences are terminal type dependent and are specified in the header section of the TDT. The string consists of a series of 8 characters followed by a terminating X'FF'. The data is positional, and the IBM 7171 expects each one to be identified. The pointer in the header block indicates the address of the first of these bytes. The order of the bytes is as follows:

1. Reset introducer

   This is the initial delimiter,if specified,for all the reset string except Pacing Start and Pacing Stop. If not specified the value will be(X'FF')
2. Master Reset
3. Character Error Reset
4. Keyboard Unlock ("3270 RESET")
5. Type-ahead Purge
6. Pacing Start
7. Pacing Stop
8. Maintenance Facility Toggle.

If the actual terminal uses the default reset keyboard character sequence, the following 7 ASCII control characters are reserved for these special functions:

X'07' Master Reset
X'12' Character Error Reset
X'14' Keyboard Unlock (3270 Reset)
X'18' Type-ahead Purge
X'11' Pacing Start (XON)
X'13' Pacing Stop (XOFF)
X'17' Maintenance Facility Toggle

4.5 ASCII Character and Function Name Tables

While building the TDT source file, it may be helpful to refer to the following tables.

4.5.1 ASCII Character Name Table

The following table assigns a unique name to each 7-bit ASCII character, which will be denoted by its hexadecimal representation. These names will be used when specifying an ASCII character in the terminal definition tables.

00	NULL	20	SPACE	40	AT	60	ACCENT
01	SOH	21	EXCLAIM	41	A	61	LCA
02	STX	22	DQUOTE	42	B	62	LCB
03	ETX	23	POUND	43	C	63	LCC
04	EOT	24	DOLLAR	44	D	64	LCD
05	WRU	25	PERCENT	45	E	65	LCE
06	ACK	26	AND	46	F	66	LCF
07	BEL	27	QUOTE	47	G	67	LCG
08	BS	28	LPAREN	48	H	68	LCH
09	HT	29	RPAREN	49	I	69	LCI
0A	LF	2A	STAR	4A	J	6A	LCJ
0B	VT	2B	PLUS	4B	K	6B	LCK
0C	FF	2C	COMMA	4C	L	6C	LCL
0D	CR	2D	MINUS	4D	M	6D	LCM
0E	SO	2E	PERIOD	4E	N	6E	LCN
0F	SI	2F	SLASH	4F	O	6F	LCO
10	DLE	30	@0	50	P	70	LCP
11	XON	31	@1	51	Q	71	LCQ
12	DC2	32	@2	52	R	72	LCR
13	XOFF	33	@3	53	S	73	LCS
14	DC4	34	@4	54	T	74	LCT
15	NAK	35	@5	55	U	75	LCU
16	SYN	36	@6	56	V	76	LCV
17	ETB	37	@7	57	W	77	LCW
18	CAN	38	@8	58	X	78	LCX
19	EM	39	@9	59	Y	79	LCY
1A	SUB	3A	COLON	5A	Z	7A	LCZ
1B	ESC	3B	SEMI	5B	LBRACK	7B	LBRACE
1C	FS	3C	LESS	5C	BSLASH	7C	BAR
1D	GS	3D	EQUAL	5D	RBRACK	7D	RBRACE
1E	RS	3E	GREATER	5E	UPARROW	7E	TILDE
1F	US	3F	QUESTION	5F	UNDER	7F	DEL

Figure 4-6. ASCII Character Name Table

4.5.2 Function Name Table

The following table identifies unique name to each function that can be invoked by pressing a key or key sequence on the keyboard of an ASCII terminal. The 3270-type and IBM 7171 extended functions correspond to internal IBM 7171 names of subroutines (entry points) which will be called from the input parse table to execute those particular functions. Figure 4-7 lists these internal names, together with a more descriptive function notation.

Routine Number	Routine Name	Routine Description
X'08'	LPRT	Echoback Local Print Sequence
X'13'	DPFM	Special Character (DUP or FIELDMARK)
X'11'	CURSEL	Cursor Select (emulate light pen)
X'0E'	RESHOW	Redisplay Screen Contents
X'12'	EINP	Erase Input
X'0C'	EEOF	Erase EOF
X'09'	DELETE	Delete Character
X'07'	INSRT	Toggle Insert Mode
X'0A'	TAB	Field Tab
X'0F'	BTAB	Field Backtab
X'1C'	COLTAB	Column Tab
X'1D'	COLBAK	Column Backtab
X'1E'	INDENT	Indent
X'1F'	UNDENT	Undent
X'02'	ATTN	Attention Generating (needs parameters for: ENTER, CLEAR, PF1 - PF36, PA1 - PA3, TREQ)
X'0B'	NL	Newline
X'0D'	HOME	Home
X'05'	UP	Cursor Up
X'06'	DOWN	Cursor Down
X'04'	RIGHT	Cursor Right
X'03'	LEFT	Cursor Left
X'17'	SETTAB	Set Column Tab
X'2A'	DELTAB	Delete Column Tab
X'19'	SETMRG	Set Left Margin
X'1B'	SETHOM	Set Home Line
X'18'	CLRTAB	Delete All Column Tabs,Home Line, and Left Margin
X'20'	SET	Set Mode of Operation (needs parameters for: z, n, e, c, v, d, i)
X'21'	RESET	Reset Mode of Operation (needs parameters for: Z, N, E, C, V, D, I)
X'14'	APLON	APL Mode On
X'16'	APLOFF	APL Mode Off
X'15'	APLEND	ASCII Input in APL Mode
X'22'	PCON	Restore Pacing
X'23'	PCOFF	Suppress Pacing
X'25'	DISC	Keyboard Initiated Line Drop
X'26'	INIT	Return to ENTER TERMINAL TYPE Message
X'27'	ALTK	Alternate Keyboard Arrangement
X'10'	DEFER	Deferred Key Execute

Figure 4-7. Function Name Table

For a more detailed description of the 3270 and IBM 7171 extended functions refer to "Chapter 3: Using ASCII Terminals" in this document. Most of the entries in Figure 4-7 are self-explanatory. Those functions requiring additional clarification are discussed briefly below:

ATTN refers to any "attention generating" 3270 key except CUR SEL. CUR SEL is given a separate assignment because, while certain kinds of light pen sensitive fields do generate attention, others only toggle the Modified Data Tag. A fourth positional parameter must be provided with CALL,ATTN to define precisely which attention generating key is being emulated.

DEFER is used when several keys generate sequences in which the beginning and ending characters are common and a middle key is the significant discriminator. An example of this are the IBM 3101 native PF keys (Alternate functions of keys on the numeric pad). They all send a three character sequence beginning with ESC and end with one of four optional change-of-direction characters (of which normally <CR> will be chosen). The IBM 7171 knows what function is being requested when the middle character arrives, but it must wait until the trailing character is received. An example is provided in "Input Parse Table" on page 4-35.

DPFM allows a control key sequence to be turned into a special graphic character. There are two such 3270 characters, DUP and FM, which act in many ways like graphic characters but are transmitted to the host as control characters. When a "CALL DPFM" statement is encountered, the PARAM field contains a one-byte value of X'80' for the DUP character, and X'7F' for the FM character stored in the current screen image in the current cursor position, subject to all the rules for protected fields and insert-mode that apply to graphic characters.

SET

RESET are used to turn one of 16 special setup bits on and off. The bit number (0-15) is provided as the parameter to the SET and RESET functions. See "4.5.3 Parameters to Pass to IBM 7171 Routines" on page 4-46 for a list of the parameters. Refer to "Chapter 3. Using ASCII Terminals on the IBM 7171" on page 3-1 for a more detailed description of these functions.

PCON

PCOFF control recognition of the pacing characters defined by the RCHRS string. When pacing characters are not recognized, they are treated as normal input.

ALTK selects an alternate keyboard arrangement based on the parm value. Two keyboard arrangements are defined. When the parm is 0 (or is omitted), the standard typewriter (QWERTY) keyboard arrangement is selected. When the parm is 1, the Dvorak keyboard arrangement is selected. Refer to "Chapter 3. Using ASCII Terminals on the IBM 7171" on page 3-1 for a more detailed description.

4.5.3 Parameters to Pass to IBM 7171 Routines

Following is a list of all possible parameters to pass to IBM 7171 routines:

Routine	Parameter	Meaning
DPFM	X'80' X'7F'	Character is a DUP Character is a FM
ALTK	1 0	Select DVORAK keyboard Select QWERTY keyboard
DEFER	routine	The calling of a routine was deferred. The parameter indicates the routine that was intended.
ATTN	1 2 3 4 6 8 9 10 11 12 46	PA1 was the ATTN Generating Key PA2 PA3 Clear Test Request Cursor Select Status Message Identifier Enter PF 1 PF 2 .... PF 36
SET	0 1 2 3 4 8 9	Enable Zones Mode 3270 Null Processing Reverse Enter/Newline Keys Reverse Column/Field Tab Keys Allow Alphanumeric Input in Numeric Only Fld Alternate Attribute Display 3278 Insert Mode
RESET	0 1 2 3 4 8 9	Disable Zones Mode Improved Null Processing Restore Enter/Newline Keys Restore Column/Field Tab Keys Disallow Alphanumeric in Numeric Only Field Primary Attribute Display 3277 Insert Mode

Figure 4-8. Possible Parameters to Pass to Routines

4.5.4 ROM Pointers to Internal Information

At the start of ROM, there is a list of addresses to all of the standard IBM 7171 information. For a description refer to "Appendix F. ROM Data Base Organization." The user is advised to use the information found here as often as possible. The addresses can be found using the Maintenance Facility, described in Chapter 9.

Index	Address	Pointer
8000	4000	Address of IBM3101 TDT
8001	4002	Reserved for future use
8002	4004	Address of TVI912 TDT
8003	4006	Address of TVI920 TDT
8004	4008	Address of TVI950 TDT
8005	400A	Address of TVI950R TDT
8006	400C	Address of ADM31 TDT
8007	400E	Address of ADM3A TDT
8008	4010	Address of VT100 TDT
8009	4012	Address of DM1520 TDT
800A	4014	Address of DM1521 TDT
800B	4016	Address of DM3045 TDT
800C	4018	Address of TYPETERM TDT
800D	401A	Address of HARDCOPY TDT
8021	4042	Address of standard Reset Character String
8070	40E0	Address of list of Terminal Translate Tables
8071	40E2	Address of list of Host Translate Table Pointers
8072	40E4	Address of Host Read Translate Table
8073	40E6	Address of Input ID Compound Formation Table
8074	40E8	Address of 3277 Output Basic Table
8075	40EA	Address of Output ID compound Analysis Table

Figure 4-9. ROM pointers

4.6 Modifying/Adding TDTs Using the Maintenance Facility

The instructions given in this section explain the use of the Maintenance Facility to install a TDT. It is the installer's responsibility to generate the correct TDT information. As stated before, each TDT is terminal dependent. The user should refer to each terminal technical reference manual for each terminal type defined. The following list is a brief summary of the steps necessary to install a new terminal type using the Maintenance Facility:

1. Define the TDT on paper first.

   The reason the TDT should be defined on paper first is that accuracy is essential. Anything less than complete accuracy will cause unexpected and unpredictable events to occur.

   The following information is necessary for a TDT :

   a. Terminal Header Information
   b. Keyboard Input Strings (Input strings from a terminal)
   c. Reset Character String (An input string from terminal)
   d. CSS pointers and Strings (Output strings to a terminal)
   e. Graphics Strings and Highlighting.

2. Refer to Chapter 9 for instructions for using the Maintenance Facility. The Maintenance Facility will be the tool used for making changes to NV-RAM.

3. Insert the terminal name into the terminal names information area.

4. Insert the TDT into the NV-RAM.

5. Update DC00:0002 with the next available address in the NV-RAM.

4.6.1 Terminal Names List

The terminal names list is a group of records, which may be arranged sequentially or in a linked list. In general, the terminal names list initially resides at the beginning of the Terminal Definition Table Area, DC00:0300. The terminal names list may be extended by a transfer field in a terminal name record if the next sequential storage area has been filled. It is the user's responsibility to ensure that the terminal names list does not contaminate other data areas. To allow expansion of the names list, the user should leave room for a transfer field at the end of the terminal names area, 2 bytes. There are initially 12 terminal name records in the TDT Area. These terminals are described in "Appendix B. IBM 7171 Supplied Terminal Definition Tables." If the TDTs described do not include the definitions of all the terminals that will be connected with to IBM 7171, then the user will need to expand the terminal names list and add the appropriate TDTs.

To expand the terminal names list, the following must be done:

1. Find the end of the terminal names list. The user must scan the entire terminal names list which begins at DC00:0300. The user will need to understand the format of terminal name records (see below).

2. Determine the storage needed for new records. The user must know how many bytes of storage are required for the additions. Each ASCII character requires 1 byte, and each pointer or optional field requires 2 bytes.

3. Now allocate the storage. The address of the first available byte can be found at address DC00:0002. To allocate the storage, add the number of bytes computed in the step 2, to the value found at DC00:0002. Remember these are hex values. Write out the sum at address DC00:0002.

4. Now enter the terminal names records. Here is a forward reference. The user needs to know the address of the first byte for each new TDT. It is up to the user to remember to fill in these values when the values become known. If all the entries are first written out on paper it is possible to derive the addresses the user will need at this point for the terminal names list.

5. Make sure the terminal names list ends with 2 bytes of zeroes, X'0000'.

Terminal Name Record Format

The format of the terminal name record appears below.

Terminal Name (8 bytes, Left Justified, Blank-filled)	Pointer to TDT (2 bytes)	Optional (This field is variable length)

The terminal name record has three basic fields. These are:

Terminal Name
The terminal name field is 8 uppercase ASCII characters long, left justified, and blank filled.

TDT Pointer
The second field is a word (2 bytes) used to store the pointer of the first byte of the Terminal Definition Table. All tables should begin on even byte addresses. This will make memory accesses more efficient.

Optional
The optional field can have 0, 1, or more words (2 byte). Each word in this field has the Most Significant Bit (MSB) equal to 1, with the exception that the list terminator has a value equal to X'0000'. The uses of these words are:

1. Signal that a carriage return is to be performed after this record's terminal name is printed. X'FFFF' is the code which identifies this use.

2. Signal that all terminal names following this record are to be hidden. X'FFFE' is the code which identifies this use.

3. Identify the address of the next record in the terminal names list. For this use the high bit equal to 1, and the High Order byte is not equal to X'FF'. The address of the next record will be the 7 low order bits of the high order byte concatenated with the Low Order Byte (i.e. the hex value of the two bytes minus X'8000').

   For Example suppose the Optional Carriage Control word equals:

   83F4 - This means the next record can be found at address DC00:03F4

4. Terminate the terminal names list. This must be the last member of the terminal names list. X'0000' is a the code which identifies this use.

Examples of Terminal Name Records

These are hypothetical examples of terminal names records:

1. ESSTTY1 9A03FFFF

   - The terminal's name is ESSTTY1
   - The terminal's TDT begins at DC00:039A
   - Since the last two bytes are X'FFFF'there will be a Carriage Return after ESSTTY1 is printed.

2. T3A94 F204FFFE

   - The terminal's name is T3A94
   - The terminal's TDT begins at DC00:04F2
   - Since the last two bytes are X'FFFE'any terminal names after this record will be hidden.

These are more examples of terminal names records. These examples, explain the use of the Maintenance Facility to retrieve the terminal name records from NV-Ram.

1. Here is an example of a fictitious terminal name record which happens to be the last record in the terminal names list. The terminal name is IBM3101 and we assume that the record begins at DC00:0350.

   ```
   Enter:m DC00:0350 C

      Outputs this string:

      DC00:0350  4942 4D33 3130 3120 0080 0000
   ```

   Which the user should interpret as follows:

   - X'DC00'=the segment address specified
   - X'0350'=the offset address specified
   - X'49'=I
   - X'42'=B
   - X'4D'=M
   - X'33'=3
   - X'31'=1
   - X'30'=0
   - X'31'=1
   - X'20'= a Blank (SP)
   - X'8000'= an address pointing to the first byte of the address containing the TDT (Remember word are stored byte reversed. Addresses and pointers are words)
   - X'0000'= End of terminal names list.

   Here the terminal name is IBM3101, left justified, blank filled. The TDT begins at the address which is at DC00:4000, and the optional field is a flag denoting the end of the terminal names list.

2. Here is another example of the terminal names list. The terminal name record for an ESAT1 is followed by the record for ESAT2. After each name has been printed on the same line, there is a carriage return and a transfer field to the next record at DC00:045C. The record begins at DC00:036A for this example:

```
Enter:m DC00:036A 14

Outputs this string (24 bytes, X'14' bytes):

DC00:036A  4553 4154 3120 2020 A20F 4553 4154 3220 2020 DC0F FFFF 5C84
```

Which is interpreted as follows:

- X'DC00'=the segment address specified
- X'036A'=the offset address specified
- X'45'=E
- X'53'=S
- X'41'=A
- X'54'=T
- X'31'=1
- X'20'= a Blank (SP)
- X'20'= a Blank (SP)
- X'20'= a Blank (SP)
- X'0FA2'= the address of the first byte of ESSAT1's TDT
- X'45'=E
- X'53'=S
- X'41'=A
- X'54'=T
- X'31'=2
- X'20'= a Blank (SP)
- X'20'= a Blank (SP)
- X'20'= a Blank (SP)
- X'0FDC'= the address of the first byte of ESSAT2's TDT
- X'FFFF'= Carriage Return after printing out terminal name
- X'845C'= The X'045C' is the address of the next record in the terminal name list.

These terminal name records define two terminals, ESAT1, and ESAT2. The optional fields(i.e. the last two words) provide a flag indicating a carriage return should be printed after the terminal names, and a link to the remainder of the terminal names list.

Note: When a transfer address is used in the optional filed only bit 0 thru 14 can be used for the address. Bit 15, the most Significant Bit, equal to 1 identifies this is an optional field. The user must remember to set the MSB of the pointer if it is necessary to extend the terminal names list.

Examples of a Terminal Names List

The following are examples of terminal names list:

TNX - Denotes terminal name record X, where X equals some integer.

Example 1:

TN1	TN2	TN3

DC00:0300

For example 1, There are 3 terminal names records. Each defines a terminal name. The third record TN3, also terminates the terminal names list.

Example 2:

TN1	TN2	TN3	.....	TN9	TN10	TN11	TN12

DC00:0300

TN13	TN14	TN15

DC00:05CB

For example 2, there are 15 terminal name records. TN12 has a list transfer field which points to TN13. TN12 looks like this:

```
58595A3130302020CC05FFFFCB85
```

The terminal's name is XYZ100. Its TDT begins at DC00:05CC. After the terminal name is printed, a carriage return is printed. TN12's last two bytes are the address of the first byte of TN13 plus X'8000'. TN15 is the end of the list. TN15 looks like this:

```
4142433931202020D20CFFFE0000
```

This terminal's name is ABC91, its TDT begins at DC00:0CD2. All terminal names which appear after this record are hidden because of the optional carriage control sequence of X'FFFE'. The X'0000' at the end of TN15 denotes the end of the terminal names list.

How to Find the End of the Terminal Names List.

Each terminal name record consists of an 8 byte name field, a 2 byte address, and 0, 1, or more optional fields. The optional field(s) provide information that will be used when listing out the terminal names list, and it will provide the next record address for linking the terminal names list.

All 2 byte fields are written as a word, using 'sw', and read with a 'mw' command. All 8 byte fields are written as a string of bytes using the 's' command, and read with a 'm' command. The 2 byte fields written by a 'sw' command may be read with a ‘m’ command, but it is important the user understands byte reversal. When a word is written out, the low order byte is stored in the low address, and the high order byte in the high address. This is byte reversal. For example:

```
sw DC00:039A 1234
```

The number, X‘1234’, is written as a word so the bytes appear to be reversed in memory when read from low address to high address, which is what occurs when reading a string of bytes. Here is an example of reading a string of bytes.

```
m DC00:039A 2
```

The ‘m’ command will read 2 bytes at DC00:039A. The value stored at X‘039A’ is X‘34’, and the value stored at X‘039B’ is X‘12’. The ‘m’ command will display the bytes in the order they exist in memory. The output of the ‘m’ command is:

```
DC00:039A 3412
```

When using the ‘mw’ command, the byte reversal is transparent. Here is an example of the command to read the word and its output:

```
mw DC00:039A
```

```
DC00:039A 1234
```

The user may find the end of the terminal names list by scanning the terminal names list which begins at address DC00:0300 for the end of list marker, X‘0000’. The first 8 bytes will be the hex codes for the 8 ASCII characters which define the terminal name. The next two bytes address the TDT of the terminal whose name has been defined by the 8 preceding bytes. The next two bytes may be:

1. The first two bytes of the next terminal name.
2. One of the following 2 byte fields:
   a. The end of terminal names list flag, X‘0000’.
   b. An address which defines where the next terminal name record may be found.
   c. A flag denoting the remaining terminal names are ‘hidden’.
   d. A flag denoting there should be a Carriage Return.

There may be more than 1 - 2 byte field following the TDT’s address. Each of the 2 byte field(s), except for the list terminator, X‘0000’, has a MSB equal to 1. This will help the user distinguish between terminal names and optional fields because ASCII characters have bytes with values less than X‘80’.

When the user encounters a link, X‘8000’ should be subtracted from the value of the transfer field. When a transfer occurs, continue scanning the terminal names list from the calculated address.

Examples of Entering Terminal Name Records

This section section provides examples of entering terminal name records:

1. Suppose the end of the terminal names list is at DC00:04A6 The terminals name is ESS3A94, its TDT begins at DC00:059A, and we want a carriage return after the name is printed.

Here are the Maintenance Facility commands to enter TNX:

```
s DC00:04A6 'ESS3A94 '
sw DC00:04AE 059A
sw DC00:04B0 FFFF
sw DC00:04B2 0000
```

The user can look at the terminal names list we just created by entering:

```
m DC00:04A6 E
```

M is a memory read command, X'E' is the number of bytes to display starting at address DC00:04A6. The Maintenance Facility returns hex codes. In the example below a translation has been done immediately below the hex code for the reader's convenience.

```
4553 5333 4139 3420 9A05 FFFF 0000
----------------------------------
E S  S 3  A 9  4 SP 059A FFFF 0000
```

Notice 'sw' stores words byte reversed, that is the low order byte at the low address, ie. if 059A is stored as a word, in memory it looks like X'9A05'. If the user writes a word using the 'sw' command, the 'mw' command will read the word correctly, ie.

```
m DC00:04A6 2      will output 9A05
mw DC00:04A6       will output 059A
```

2. Suppose the end of the terminal names list is at DC00:063E, but there is not enough storage available in the terminal names record area to add another record. The following explains the steps necessary to add more terminal name records. First, find the address of the next available storage location. This is performed by checking at value at location DC00:0002, which contains this information. Check the content of DC00:0002 with this command:

   ```
   mw DC00:0002
   ```

   Which outputs:

   ```
   DC00:0002 066C
   ```

   The address X'066C' defines where the next terminal name record will be written. However, before the new record can be added, a link must be added. The link will be a list transfer field written over the previous end of terminal names list marker, X'0000'. To generate the proper link, add X'8000' to the address of the record being linked, X'8000' + X'066C' = X'866C'. Now write the list transfer field using the command:.

   ```
   sw DC00:063E 866C
   ```

   This establishes a link to the new record. Now write out the new record or records. Be sure to terminate the terminal names list with a X'0000'. Finally update the next available address (this is the value stored at DC00:0002). In this case this value should be the address of the last byte in the terminal names list plus one.

4.6.2 Terminal Definition Tables, TDTs:

This is an description of the TDTs which reside in NV-RAM. The description uses a pseudo language which is defined in the next section. The description is presented as an aid in generating TDTs. The pseudo definition language description is further developed by several block diagrams on the following pages. The definitions proceed in a top down fashion for both the pseudo definitions and the block diagrams.

Symbol Definitions for Pseudo Definition Language

¬	- a pointer to the data item described, a one word address
;	- a statement separator,
=	- equivalence
|	- the select indicator, chose from the items on either side
X	- Hex Constant Prefix, ie. X'6B'
'	- Constant delimiter, open and close string, ie. 'a'
//	- Concatenation operator, ie. 'a'//'b' = 'ab'

Key Words and Their Meanings for the Pseudo Definition Language

RECORD	- An Aggregate of different related data objects
END	- Terminator for the RECORD Key word
BYTE	- a Unit of storage consisting of 8 bits
WORD	- a Unit of storage consisting of 16 bits

Variable Naming Conventions

PTR indicates the variable is an address pointing to storage. BLOCK indicates the variable is of varying length, depending on the application. The variable names in the type definitions are equivalent to the variables described in this manual. To get a complete understanding of each variable name, the user may need to refer back to different sections.

```
Type

TDT_TYPE = TDT_STORAGE_AREA;

  TDT_STORAGE_AREA = RECORD
        TERMINAL_HEADER = HEADER_RECORD;
        CSS_POINTERS    = CSS_PTR_REC;
        CSS_STRINGS     = BLOCK_OF_STRINGS;
        INPUT_PARSE_TBL = BLOCK_PARSE_TABLE_TYPE
   END;

         HEADER_RECORD = RECORD
            STATUS_FLAG1                        =  BYTE;
            STATUS_FLAG2                        =  BYTE;
            PTR_PARSE_TABLE                     =  ¬PARSE_TABLE_TYPE;
            PTR_1ST_CSS_PTR                     =  ¬CSS_PTR_REC;
            CURSOR_ORIGIN                       =  CURSOR_ORIGIN_TYPE;
            PTR_TO_HOST_TRANSLATE_TABLE_ADDRESS =  ¬HOST_TRANSLATE_TABLE;
            PTR_TO_TERM_TRANSLATE_TABLE         =  ¬TERM_TRANS_TABLE;
            PTR_TO_RESET_STRING                 =  ¬RESET_STRING;
            DELAY                               =  DELAY;
            PTR_TO_SET_GRPK_RENDITION_STRING    =  ¬GRPK_REDITION_STRING
          END;

         CSS_PTR_REC = RECORD
           PTR_CURSOR_REPOSITION_STRING     =  ¬STRING;
           PTR_ERASE_END_OF_LINE            =  ¬STRING;
           PTR_LOCAL_PRINT                  =  ¬STRING;
           PTR_TONE                         =  ¬STRING;
           PTR_CURSOR_LEFT                  =  ¬STRING;
           PTR_CURSOR_RIGHT                 =  ¬STRING;
           UNUSED1                          =  ¬STRING;
           UNUSED2                          =  ¬STRING;
           PTR_SIGNAL_INSERT_MODE_ON        =  ¬STRING;
           PTR_SIGNAL_INSERT_MODE_OFF       =  ¬STRING;
           PTR_DISCONNECT                   =  ¬STRING;
           PTR_CLEAR                        =  ¬STRING;
           PTR_TERMINAL_INITIALIZE          =  ¬STRING;
           PTR_ILLEGAL_ASCII_CHARACTER      =  ¬STRING;
           PTR_ILLEGAL_APL_CHARACTER        =  ¬STRING;
           PTR_APL_CHARACTERS_ON            =  ¬STRING;
           PTR_APL_CHARACTERS_OFF           =  ¬STRING;
           PTR_DISPLAY_MODE                 =  ¬STRING;
           UNUSED3                          =  ¬STRING
         END;

    BLOCK_OF_STRINGS = STRING | STRING // BLOCK_OF_STRINGS;

       STRING = RECORD
           SEQUENCE       = ASCII_STRING;
           SEQ_TERMINATOR =  X'FF'
         END;

    ASCII_STRING =  ASCII_CHARACTER   |  ASCII_CHARACTER // ASCII_STRING;

    BLOCK_PARSE_TABLE_TYPE = PARSE_TABLE_NODE |
                              PARSE_TABLE_NODE//BLOCK_PARSE_TABLE_TYPE;

       PARSE_TABLE_NODE = RECORD
          CHARACTER = BYTE;
          PARAMETER = BYTE;
          TARGET    = WORD
        END;
```

TDT_TYPE

TERMINAL HEADER	18 decimal bytes
CSS POINTERS	38 decimal bytes
CSS STRINGS	VARIABLE -- dependent on type of terminal
INPUT PARSE TBL	VARIABLE -- dependent on type of terminal

The length of the TDT depends upon the type of terminal being used. However, all TDTs have a minimum fixed length of 56 decimal bytes.

TERMINAL HEADER -- Fixed length of 18 decimal bytes

STATUS FLAG1 - 1 byte	STATUS FLAG2 - 1 byte
PTR PARSE TABLE	2 bytes
PTR 1ST CSS PTR	2 bytes
CURSOR ORIGIN	2 bytes
PTR TO HOST TRANSLATE TABLE ADDRESS	2 bytes
PTR TO TERM TRANSLATE TABLE	2 bytes
PTR TO RESET STRING	2 bytes
DELAY	2 bytes
PTR TO SET GRPK RENDITION STRING	2 bytes

CSS Pointers—Fixed length of 38 decimal bytes.

PTR CURSOR REPOSITION STRING	2 bytes
PTR ERASE END OF LINE	2 bytes
PTR LOCAL PRINT	2 bytes
PTR TONE	2 bytes
PTR CURSOR LEFT	2 bytes
PTR CURSOR RIGHT	2 bytes
UNUSED1	2 bytes
UNUSED2	2 bytes
PTR SIGNAL INSERT MODE ON	2 bytes
PTR SIGNAL INSERT MODE OFF	2 bytes
PTR DISCONNECT	2 bytes
PTR CLEAR	2 bytes
PTR TERMINAL INITIALIZE	2 bytes
PTR ILLEGAL ASCII CHARACTER	2 bytes
PTR ILLEGAL APL CHARACTER	2 bytes
PTR APL CHARACTERS ON	2 bytes
PTR APL CHARACTERS OFF	2 bytes
PTR DISPLAY MODE	2 bytes
UNUSED3	2 bytes

CSS—String lengths are variable.

CURSOR REPOSITION 'FF'	ERASE END OF LINE 'FF'	LOCAL PRINT 'FF'	
TONE 'FF'	CURSOR LFT 'FF'	CURSOR RT 'FF'	SIGNAL INS MODE ON 'FF'
SIGNAL INS MODE OFF 'FF'	DISCONNECT 'FF'	CLEAR 'FF'	
ILLEGAL_ASCII_CHARACTER 'FF'	ILLEGAL_APL_CHARACTER 'FF'		
APL_CHARACTERS_ON 'FF'	APL_CHARACTERS_OFF 'FF'	DISPLAY_MODE 'FF'	

Input Parse Table.

PARSE TABLE NODE 1 4 bytes
PARSE TABLE NODE 2 4 bytes
. . . . PARSE TABLE NODE (i) . each node is 4 bytes . . .
PARSE TABLE NODE N 4 bytes

PARSE_TABLE_NODE_I

CHARACTER 1 byte	PARAMETER 1 byte	TARGET 1 word

Entering a TDT

1. Enter the Maintenance Facility Described in Chapter 9.

2. Get the address of the next available storage area in NV-RAM. This can be accomplished by using the Maintenance Facility command:

   ```
   mw DC00:0002
   ```

   Maintenance Facility will respond by printing out the contents of the memory location at DC00:0002. This is the address of the first available byte in NV-RAM. After the TDT has been entered, DC00:0002 must be updated to the first byte available. Normally the first byte available will follow the last byte of the TDT most recently entered into the NV-RAM.

3. Now allocate the terminal header area. This area is of fixed length, 18 decimal bytes. 2 bytes for flags, and 8 words for pointers and parameters.

4. Enter the terminal header information (18 bytes decimal)

   Perform the following 10 steps, substituting the last parameter in each command with the described value, and performing the arithmetic before entering the command (ie. 04E2h+4, enter 01E6h in place of 04E2h+4) (04E2 is the examples starting point. Use the value found at DC00:0002 for the user's starting point).

   Refer to this chapter for descriptions of each terminal header parameter value.

   a. s DC00:04E2 Status_flag_1

   b. s DC00:04E2+1 Status_flag_2

   c. sw DC00:04E2+2 PTR_to_Parse_table_entry_address

   d. sw DC00:04E2+4 PTR_to_1st_CSS_PTR

   e. sw DC00:04E2+6 PTR_to_Cursor_Origin

   f. sw DC00:04E2+8 PTR_to_Host_Translate_Table_Address

   g. sw DC00:04E2+10d PTR_to_Terminal_translate_table

   h. sw DC00:04E2+12d PTR_to_Reset_Character_String

   i. sw DC00:04E2+14d Delay

   j. sw DC00:04E2+16d PTR_to_Set_GRPK_Rendition_STR

5. Now allocate the CSS Pointer Area. This Area is of fixed length, 38 bytes. 19 words for pointers to Control Sequence Strings.

   There are 19 CSS Pointers.

   Since storage is sequentially allocated, we can continue building our TDT after the terminal header area. The 19 CSS Pointers will use the next 38 decimal bytes, 2 bytes per pointer.

   The CSS Pointers should be entered using the 'sw' command for example:

   ```
   sw DC00:TUVW address
   ```

6. Enter the CSSs

   The Strings pointed to by the 19 CSS Pointers will stored after the CSS Pointer area. It is important to keep an accurate count of every byte the user allocates as the user enters the Control Sequence Strings.

For example, the first CSS Pointer, points to a string of characters that the terminal recognizes as the Cursor Reposition Sequence. If we continue with our example from the terminal header area, we can see that bytes 04E2 thru 04E2+17d have been used by the terminal header area. The next address, 04E2+18d will be the first CSS pointer, which is a pointer to the Cursor Reposition Sequence. The Cursor Reposition sequence will be stored in the first available area, which is after the CSS Pointer Area, 04E2+18d+38d. The Cursor Reposition Sequence is terminated by a byte whose value is X'FF'. The second pointer in the CSS area is pointing to the Erase EOL sequence. The Erase EOL sequence is stored in the next available area, which is after the X'FF' byte which was used to terminate the Cursor Reposition Sequence. The pointer to the Erase EOL sequence will be stored at 04E2+20d, and the actual character sequence representing Erase EOL will be at 04E2+18d+38d+(the length in bytes of the Cursor Reposition Sequence + the length of the terminator).

Please note the Control Sequence String Terminator, X'FF', must be used to terminate each CSS. Without the terminator, the IBM 7171 will not output the correct string sequence to a terminal.

While the CSS Pointers should be entered using the 'sw' command the CSSs should be entered using the 's' command of section 9.22. For example suppose we want the sequence:

```
ESC G @
```

Look up the ASCII Hex values for ESC, G, and @. The S command will store the string properly when entered as follows (XYZA is the appropriate offset)

```
S DC00:XYZA 1B47 40FF
```

Note the last byte X'FF' was the CSS terminator.

Enter the Input Parse Table: The Input Parse Table is the chief tool for decoding and correlating terminal keystrokes. Each keystroke or keystroke sequence is translated into a request for some specific action such as cursor movement, clear screen, home, etc. The format of the Input Parse Table is an aggregate of nodes. Each node define the character, and the action that should take place. Refer back to "Input Parse Table" on page 4-35 for a more detailed description of the Input Parse Table.

The Input Parse Table may follow the CSS string area. Once you know the address of the first byte, you can fill in the Input Parse Table pointer which is in the TDT Terminal Header Record. Then you should fill in each node in the Parse Table which is exactly 4 bytes long. Each byte in the node must be specified. It is strongly recommended that the entire parse table be described on paper first. This will allow you to proceed with each entry of the table without having to keep track of unknown addresses which won't be known until latter in the parse table description.

Example of Entering a Parse Table Node: For demonstration purposes, we will describe the entering of one node of an Input Parse Table. Assume that you want to enter at the location DC00:07A2 the node "ESC GOTO IBMESC." Then the sequence of operator commands to accomplish this is:

```
s   DC00:07A2   1B          ;ESC=X'1B'
s   DC00:07A3   00
sw  DC00:07A4   5372        ;This the GOTO address IBMESC
```

This node is equivalent to the first node in Figure 4-4 on page 4-38. Note that this figure contains a byte for byte representation of the 7171 memory and that the GOTO address is stored in reverse byte notation. Reverse Byte notation is simply the low order byte is placed in the low address, the high order byte in the address specified plus one. The sw command of MAINTENANCE, stores words in reverse byte notation automatically. Thus it is recommended that all addresses, and pointers should be written and read, using Maintenance's word instructions (these commands are two letters with 'w' being the second letter, ie sw, mw).

Whenever an address or pointer is read in a byte by byte manner, then any bytes that where written as a word are reversed with respect to their intended order. For example we could read in the parse table node we wrote out two ways, Byte-wise, or exactly the way it was written, namely 2 bytes and a word.

For byte-wise operation issuing the command:

```
m DC00:07A1 4
```

would output, 1B00 7253.

Note: The address is retrieved byte reversal. To retrieve exactly what was written we have to issue the following sequence of commands:

```
m  DC00:07A1 2
mw DC00:07A3
```

would output 1B00 5372.

The main point here is to write all addresses and pointers using word commands. In this manner Input Parse Table nodes will be entered as easily as possible using the Maintenance Facility.

Setting Host Disconnect Methods

This section describes setting up each of the two host disconnect methods. A local 3270 terminal is always connected to its control unit by a coaxial cable. An ASCII terminal, however, can be remotely attached to the IBM 7171 over switched network telephone lines. If such a connection were lost and the host application cannot be appropriately notified, then the next user to dial into the same phone line would continue with the interrupted session. The IBM 7171 provides two methods to signal line drop to different host access methods.

Setting Up the Asynchronous Status Method: An isolated Device End is a signal from a 3270 terminal that it has just powered on. This response is enough for VM to put the session into a disconnected state. A BTAM application program can use the READYQ keyword on the DCB to intercept the Device End, but many existing BTAM applications do not use READYQ. TCAM and VTAM both ignore isolated Device Ends. If this method is chosen, the IBM 7171 will allow any status/sense combination to be sent to the host.

The word at DC00:E is the location where the status/sense information is stored. To select this option, store the desired sense in the first byte and the desired status in the second byte. If this word is zero, then this method is disabled.

For example, if the host is running VM, then an isolated Device End is sufficient to disconnect the session. Therefore, store x'00' (no sense) at DC00:E, and x'04' (Device End status) at DC00:F.

Setting Up the Send Predefined Message to Host Method: When this method is used, the IBM 7171 will generate an Attention to the host when a line is dropped, and will respond to the resulting read with a predefined message.

The word at DC00:2 contains the offset of the next available position in non-volatile RAM, assuming that it has been updated each time a change to NV-RAM has been made. This value must be stored in the word at DC00:A, which is defined to be the offset of the message to be sent to the host on line drop. By default, this offset is zero, which means no message will be sent to the host when a line drops. Stored at this offset is a word whose value is the length of the message to send followed by the message itself. This message must be in the form of an EBCDIC datastream which could normally be built by a Read Modified command, as no interpretation is performed.

For example, suppose the word at DC00:2 contains x'039A' (the byte reversed default for an unmodified system). Since there is free space starting at DC00:39A, the message will go there. Store x'039A' (byte reversed) at DC00:A. This tells the IBM 7171 to send an Attention when the line drops, and to respond to the resulting read with the message stored at DC00:39C. At DC00:39A, store x'000A' (byte reversed). This is the length of the message to send. At DC00:39C, store the following hex string:

```
x'016C6102D3D6C7D6C6C6'
```

The first four bytes of this string are a Test Request header, and the next six bytes are the EBCDIC values for word 'LOGOFF'. Finally, to update the next available position in NV-RAM, store a x'03A6' (byte reversed) at DC00:2. The effect of all this is to simulate the datastream that would be generated if on a clear screen the user typed the word LOGOFF and then pressed the Test Request key. This will generate a disconnect on ACF VTAM Version 1 Release 3 or Version 2.

Chapter 5. IBM 7171 I/O Interface to Terminals

5.1 Common Carrier Interface

The IBM 7171 is a controller through which RS-232-C ASCII terminals, printers, and similar devices can communicate with a host IBM 43xx or 308x processor. As many as 64 such devices can be attached to the IBM 7171, and this section describes the interface to which they are attached.

Each of the IBM 7171 ports is a DTE and expects to be connected to a terminal through an asynchronous modem. The modem can be connected to a common carrier's switched network, or to leased transmission lines. It is also possible to attach a terminal directly to the interface through a specially wired cable called a "Null Modem Cable."

5.1.1 United States Interface Compliance

The IBM 7171 conforms to the following standards when installed within the United States.

Functional/Electrical/Mechanical

EIA RS-232-C Interface between Data Terminal Equipment (DTE) and Data Communications Equipment (DCE) employing Serial Binary Data Interchange

The EIA RS-232-C standard defines functional, electrical, and some of the mechanical characteristics of the interchange circuits used between the terminals and the IBM 7171.

Mechanical

ISO 2110 Data Communication - 25-pin DTE/DCE interface connector and pin assignments

International Standard ISO 2110 defines the 25-pin "D" connector commonly used by RS-232-C equipment.

5.1.2 World Trade Interface Standards Compliance

The IBM 7171 conforms to the following standards when installed outside of the United States.

Functional/Procedural

CCITT V.24	List of Definitions for Interchange Circuits Between Data Terminal Equipment (DTE) and Data Circuit Terminating Equipment (DCE).

Electrical Interface

CCITT V.28	Electrical Characteristics for unbalanced double-current interchange circuits.

Mechanical Interface

ISO 2110	Data Communication - 25-pin DTE/DCE interface connector and pin assignments

5.2 RS-232-C Hardware Interface Description

This section describes the RS-232-C ports of the IBM 7171.

5.2.1 The IBM 7171 Definition as a DTE

The IBM 7171 is configured as a DTE. It drives the signals DTR and RTS, and it receives the signals DSR, CTS, RLSD, and RI. Each port has a 25 pin male "D" connector - the type normally used for RS-232-C interfaces. By using a one-to-one cable, a port can be connected directly to a modem or another device with an interface like a modem. More sophisticated users will be able to configure other cables and connector wiring to allow the attachment of other devices which do not attach directly.

5.2.2 RS-232-C Pins Used by the IBM 7171

The following figures summarize the Data Transmission Configurations and interface standards supported. The two interface standards are basically the same except for the names of the signals on the pins.

In the switched network configuration, the cable will connect the ten pins available at the IBM 7171 port connector to the respective pins of the modem connector. This is a common one-to-one cable which is available from many vendors.

In the leased line configuration, if the attached modem drives the Ring Indicator signal (pin 22), the common one-to-one cable can be used. If the attached modem does not drive the Ring Indicator signal, then at the IBM 7171 end of the cable, pin 22 must be wired to pin 7 at the connector. It is possible to configure any port for a leased line modem, which eliminates the need to condition the Ring Indicator signal at all. Section "9.8 C - Configure IBM 7171 Ports" on page 9-6 describes how to configure a port.

For the directly connected configuration, a specially configured cable is required. That cable, which may be unique to a specific terminal, will probably have pins at one end of the cable wired to different pins at the other end, and will likely have some pins at each end jumpered to other pins at that same end. It is common, for instance, to use a cable having female connectors at both ends with pin 2 at one end wired to pin 3 at the other end.

Refer to the *IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide* (Appendix B) for specifics regarding recommended cable wiring.

TABLE 1 INTERFACE TYPE RS-232-C

DATA TRANSMISSION CONFIGURATION	INTERFACE TYPE
Full-Duplex leased line with modem	A
Full-Duplex switched line with modem	B
Full-Duplex direct connection with Null Modem	C

PIN #	INTERFACE CIRCUIT	RS-232-C	INTERFACE TYPE A	B	C	DIRECTION
1	AA	Protective Ground	x	x	x	Common
7	AB	Signal Ground	x	x	x	Common
2	BA	Transmitted Data	x	x	x	Output
3	BB	Received Data	x	x	x	Input
4	CA	Request To Send	x	x	*	Output
5	CB	Clear to send	x	x	*	Input
6	CC	Data Set Ready	x	x	*	Input
20	CD	Data Terminal Ready	x	x	*	Output
22	CE	Ring Indicator		x		Input
8	CF	Received Line Signal Detector	x	x	x	Input

Figure 5-1. Data Transmission Configuration Types and RS-232-C Pin Requirements. The direction is referenced to the IBM 7171 e.g. Received Data is an input to the IBM 7171.

The signals marked with an "x" are wired between the IBM 7171 and the attached device while the signals marked with an "*" are usually jumpered within the housings on the ends of the cable.

TABLE 2 INTERFACE TYPE CCITT-V.24

DATA TRANSMISSION CONFIGURATION	INTERFACE TYPE
Full-Duplex leased line with modem	A
Full-Duplex switched line with modem	B
Full-Duplex direct connection with Null Modem	C

PIN #	INTERFACE CIRCUIT	CCITT-V.24	INTERFACE A	B	TYPE C	DIRECTION
7	102	Signal Ground	x	x	x	Common
2	103	Transmitted Data	x	x	x	Output
3	104	Received Data	x	x	x	Input
4	105	Request To Send	x	x	*	Output
5	106	Clear to send	x	x	*	Input
6	107	Data Set Ready	x	x	*	Input
20	108.2	Data Terminal Ready	x	x	*	Output
22	125	Calling Indicator		x		
8	109	Received Line Signal Detector	x	x	x	Input

Figure 5-2. Data Transmission Configurations Types and CCITT-V.24 Pin Requirements. The direction is referenced to the IBM 7171 e.g. Received Data is an input to the IBM 7171.

The signals marked with an "x" are wired between the IBM 7171 and the attached device while the signals marked with an "*" are usually jumpered within the housings on the ends of the cable.

5.2.3 Signals Driven by the IBM 7171

The IBM 7171 drives the following signals to declare that it is ready to be used.

DTR Data Terminal Ready (Pin #20)

RTS Request To Send (Pin #4)

A modem attached to one of the IBM 7171 ports should not attempt to establish a connection until the DTR control lead is driven active. Once a connection has been made, the active states of DTR and RTS will maintain that connection. During the operational phase, the IBM 7171 will monitor these input signals and will respond to their changing states. When the IBM 7171 is to disconnect the modem, DTR and RTS are driven inactive for about four (4) seconds. When the IBM 7171 port recognizes that the attached modem has disconnected, it will then redrive DTR and RTS active awaiting a new connection.

Since the IBM 7171 expects to be communicating to a terminal through a modem, a directly connected terminal will have to respond correctly to DTR and RTS, an attachment that is commonly done.

5.2.4 Interface Leads Monitored by the IBM 7171

The following four (4) leads are monitored at each of the IBM 7171 ports.

DSR Data Set Ready (Pin #6)

CTS Clear To Send (Pin #5)

RLSD Received Line Signal Detector (Pin #8)

RI Ring/Calling Indicator (Pin #22)

These signals driven by the modem reflect the states of the modem's operation and are used by the IBM 7171 to establish the connections, send and receive data and disconnect the modem.

The IBM 7171 can use the Ring Indicator to distinguish a switched network configuration from the leased line and locally attached configurations.

5.3 Transmission Mode Supported

Devices attached to the IBM 7171 ports must be Full Duplex. They must assert the Request to Send (RTS) lead when ready to establish a connection with the IBM 7171, and this signal must be held in the active state for the duration of the session.

5.4 Baud Rates Supported

The following baud rates are supported by the IBM 7171.

- 50
- 75
- 110
- 134.5
- 150
- **300**
- **600**
- **1200**
- **1800**
- 2000
- **2400**
- **3600**
- **4800**
- 7200
- **9600**
- **19200**

The highlighted baud rates are supported for "autobaud detection" described below. Any port can be assigned any of the above rates by making an entry into Non-Volatile RAM (NV-RAM). Section "9.8 C - Configure IBM 7171 Ports" on page 9-6 describes how to configure a port including setting the baud rate.

5.4.1 Autobaud Detection Option

The IBM 7171 is able to detect the baud rate of a terminal when the connection is established as long as the rate is one of the highlighted rates listed above. If a port is configured for autobaud detection, the IBM 7171 will monitor the first character transmitted by the attached terminal. The terminal must transmit a carriage return, and from that the IBM 7171 can determine the baud rate which will be used for the remainder of the session.

5.5 Autoanswer Mode

The IBM 7171 operates in an autoanswer mode. When it is available and online, it drives the DTR output control lead active at the RS-232-C interfaces. When DTR is held active, a modem with the autoanswer feature will automatically answer incoming calls to the IBM 7171.

At certain times (when the IBM 7171 is powered off for example) DTR is not driven active. Switched network modems should not answer incoming calls when DTR is inactive.

5.6 Port Initialization

5.6.1 Monitoring of Control Leads for Initializing a Port

When the IBM 7171 is powered on and running, it monitors the input signals of each port.

During the initialization phase of each port, the IBM 7171 drives DTR and RTS active. When it is ready to begin a session, the attached device must bring the proper input signals active as described in the EIA RS-232-C standard. This attached device must be either a modem or another device which appears to be a modem. When the signal states satisfy the requirements of both the IBM 7171 and the attached device, the initialization phase is complete, and the IBM 7171 will enter the operational phase.

Initialization Control Leads Sequence

By driving DTR and RTS active, the IBM 7171 notifies the attached device that it is available for use. The attached modem must use DTR to enable autoanswer mode, while RTS is used to enable the drivers in the modem.

The IBM 7171 supports three types of configurations: switched network, leased lines, and direct connection. During its initial power on sequence, the IBM 7171 references NV-RAM, to determine the type of configuration for each port. NV-RAM contains a table of information for each port, and one entry in that table defines the type of attachment. The default entry is "auto-configuration," but that can be overridden as described in section "9.8 C - Configure IBM 7171 Ports" on page 9-6. In the auto-configuration mode the IBM 7171 uses the Ring Indicator signal to recognize if the port is attached to a switched network modem. The Ring Indicator lead (pin 22) must be driven to a valid state. If the attached device does not drive the signal to a defined active/inactive level, it must be jumpered to Signal Ground (pin 7), or the configuration should be specified as described in section "9.8 C - Configure IBM 7171 Ports" on page 9-6.

5.7 Operational Phase

The operational phase is the normal running state of each of the IBM 7171 ports. It is entered when the RS-232-C control lead handshaking of the initialization phase has been completed. The operational phase is evident when the terminal keystrokes are echoed back to the display, or messages are displayed. It is possible for the port to be in the operational phase, but for various reasons the IBM 7171 is unable to communicate with the host operating system. The user will normally see one of several indications when his port enters the operational phase.

- On a port for which the baud rate and terminal type have been defined, the host operating system logo will appear if the system is available.
- If the baud rate is defined, but the terminal type is not, the message "ENTER TERMINAL TYPE" will appear.
- If the terminal type is defined, but the baud rate is not, no indication will be seen. The user will have to type the CARRIAGE RETURN key, and then the host operating system logo will appear if the host connection is up and running.
- If neither the baud rate nor the terminal type is defined no indication will be seen. The user will have to type the CARRIAGE RETURN key, and then the message "ENTER TERMINAL TYPE" will appear.

Once the IBM 7171 enters the operational phase for any port, it will begin transferring the characters and commands to and from the attached device. The RS-232-C signals are monitored and will affect the data transfer. For example, a printer may accept a line of characters at a high rate, but then will not accept any more until that line is printed. The printer will drive the CTS signal inactive when it cannot accept any more data, and will drive it active when it is ready for more characters. The IBM 7171 will react properly to those control signals.

5.7.1 Control Leads During Operational Phase

During the operational phase, the control signals are monitored. There are two normal conditions which will cause these signals to be driven inactive. The first is caused by the attached device wanting to stop the data transfer for a moment. That stopping of the data transfer is called "pacing" and is covered in the following section. The second is caused by the attached device signalling the end of a session. When DSR, CTS or RLSD is driven inactive, the IBM 7171 starts a timer for that port. If the timer runs out before that signal becomes active again, the IBM 7171 will disconnect that port. More details on disconnection are found in the section "5.10 Disconnection" on page 5-12.

5.8 Pacing

Some attached devices, such as printers, may not be able to handle characters as fast as they can be received. Some of those devices have the ability to send a character back to the transmitting unit asking it to stop momentarily while the receiving unit catches up. When it has caught up, the receiving unit can then send another character signalling the transmitting unit to continue. This form of stopping and starting the transmitting unit on demand is called "pacing," and such devices can be used with the IBM 7171.

5.8.1 XON/XOFF

Pacing Enable

Although the IBM 7171 will receive any pacing characters sent by the attached device, it will not respond to them unless pacing recognition has been enabled. Pacing character recognition is enabled with the "Restore Pacing" setup function, and it is disabled with the "Suppress Pacing" set up function. These two functions are described in section "3.1.5 Setup Functions" on page 3-3.

XON & XOFF Characters

When pacing is enabled for a port, all characters received by the IBM 7171 on that port are compared to the defined XON and XOFF characters for that port. Different ports may have different characters for XON and XOFF.

When the attached device sends the XOFF character, and pacing is enabled, the IBM 7171 will stop sending characters to that attached device. If a second XOFF character is received, it will just be ignored.

When the attached device sends the XON character, and pacing is enabled, the IBM 7171 will resume sending any buffered characters to that attached device. If a second XON character is received by the IBM 7171 on that port, it will be ignored, and a BEL character will be sent to that attached device.

5.8.2 Control Signal Pacing

As mentioned above, the attached device can drive CTS, DSR, or RLSD inactive which stops any data transmission. Normally the CTS signal is used as a pacing control signal, and will cause the IBM 7171 port to stop transmitting momentarily. There are time limits associated with each of these control lines, and exceeding those limits will cause the IBM 7171 to disconnect that port. CTS can be inactive for as long as ten seconds, but DSR is limited to 100 milliseconds and RLSD is limited to 500 milliseconds. The CTS value is fixed, but the DSR and RLSD values are default values in the port table in NV-RAM, and are alterable as described in section "5.10 Disconnection" on page 5-12.

5.9 Attachment Configurations Supported.

The IBM 7171 supports three types of attachment configurations:

1. Switched Network
2. Leased Line
3. Locally Attached.

The configuration for each port is stored in NV-RAM and can be modified as shown in section "9.8 C - Configure IBM 7171 Ports" on page 9-6. A fourth configuration is called auto-configuration. In that mode the IBM 7171 will attempt to determine which of the above three modes to use for that port.

For the auto-configuration mode to work properly, the Ring Indicator lead (pin 22) at the IBM 7171 RS-232-C port must be properly conditioned. It must not be left floating. If the modem or terminal does not drive the Ring Indicator to the marking condition (except to indicate a ringing state at which time it is driven to the spacing condition), then that signal must be jumpered to Signal Ground (pin 7) at the IBM 7171 end of the cable. Because the receiver input impedence is high, noise can be easily coupled into a floating input which can cause the IBM 7171 to disconnect that port as described in the paragraph on invalid interface lead sequences on page 5-14.

5.9.1 Switched Network

General Description of Switched Network Configuration

The switched network is the basic dial-up telephone network used with normal telephone calls. A port attached to the switched network through a modem would have a telephone number. A user at a terminal prepares the terminal and then dials the telephone number of the modem on the IBM 7171 port. The signals on the IBM 7171 port allow the modem to answer the call and signal the IBM 7171 when the line is ready.

Cabling Requirements

To attach a modem (DCE) to the IBM 7171 port (DTE), a common one-to-one RS-232-C cable with a male connector on the DCE end and a female connector on the DTE end is used. That cable can be built from components or can be purchased from many vendors. A description of the cable is included in Appendix B of *IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide.*

General Switched Modem Specifications

Modems interfacing between the IBM 7171 port and the switched telephone network should be configured to automatically answer incoming calls. They should also be configured (if possible) to answer incoming calls only when the DTR signal is driven active by the IBM 7171 port. When DTR is not active, the IBM 7171 port is not ready to receive any calls.

Modems should be configured so the RLSD signal reflects the true state of the telephone carrier signal. If the remote modem is powered off for any reason, the IBM 7171 port should be able to detect it and react accordingly.

The Ring Indicator signal is expected to be driven by the modem even if it is configured for the autoanswer mode. The IBM 7171 expects that the Ring Indicator signal will be driven active and then inactive at least once before the line is answered.

5.9.2 Leased Line

General Description of the Leased Line Configuration

A leased line is a direct connection between two modems. The lines are not routed through the telephone company switching equipment, but rather are directly routed. They may be telephone wires through the phone system, or they may just be a twisted-pair of wires within a building. A modem attached directly to another sequences its RS-232-C interface signals differently from a modem attached to the switched network. The Ring Indicator, for example, is never driven active by a modem on a leased line. Those differences are important to the IBM 7171 as it initializes a session and as it disconnects a modem. The modems connected to leased lines are frequently very different from those used in the switched network. The switched network modems typically have a maximum baud rate of 1200 baud with many much slower than that. The quality of the telephone lines between the modems is controlled by the equipment in the phone company central office. Since leased line modems are connected directly by wires, there is no equipment in the lines to help make the quality of those lines good. Most of the leased line modems have the electronics to condition the transmission lines to get the best quality possible. With those lines properly conditioned and with more sophisticated data encoding schemes, some of these modems can pass data at 9600 baud or even faster. Part of the initialization sequence between the two modems when they are powered on is the conditioning of the lines. The IBM 7171 uses the differences between switched and leased line modems to determine how to sequence the RS-232-C signal lines.

Cabling Requirements

To attach a modem (DCE) to the IBM 7171 port (DTE), a common one-to-one RS-232-C cable with a male connector on the DCE end and a female connector on the DTE end. That cable can be built from components or can be purchased from many vendors. A description of the cable is included in Appendix B of *IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide.*

Warning: For auto-configuration to work properly, the Ring Indicator lead (pin 22) must be properly conditioned. If the modem does not drive Ring Indicator to the marking condition, then that signal must be jumpered to Signal Ground (pin 7) at the IBM 7171 end of the RS-232-C cable.

5.9.3 Locally Attached (Direct Connection)

General Description of Local Attach Configuration

The RS-232-C interface was designed to allow connection between DCE's (modems) and DTE's (terminals and computers). The RS-232-C standard allows up to 2500pf of capacitance in the signal lines which means that the cable could be up to about 50 feet long. Many users have terminals within 50 feet of their processors and do not see a need for buying two modems just to go that distance. The problem then is that they want to attach a DTE (terminal) to another DTE (computer), and that is not included in the RS-232-C standard. It has been determined that by wiring the RS-232-C cable between the DTE's in a special manner, the cable can successfully emulate a DCE. In that cable, some signals are just wrapped back to other pins on their connectors, while other signals are crossed from one end of the cable to the other. Such a cable is commonly referred to as a "null modem cable."

Users have also realized that 50 feet is a somewhat arbitrary distance. While they are in violation of the RS-232-C standard, they have been able to use cables which are much longer than the standard allows and have been able to run successfully. It is not uncommon to find cables several hundred feet long, especially when running at lower baud rates. The long cables degrade the shape of the data signals which will potentially cause more transmission errors.

Cabling Requirements

The null modem cable will be specific to each computer and terminal pair although many are the same. Each cable will have to be assembled by the user since they are not available from vendors.

A formal description of these cables is found in Appendix B of *IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide.*

Warning: For auto-configuration to work reliably, the Ring Indicator lead (pin 22) must be jumpered to Signal Ground (pin 7).

5.10 Disconnection

When the IBM 7171 terminates a session, it disconnects or drops the line. There are two classes of disconnection. In the first, the host computer has initiated the disconnection. The user may have given the host the command to logoff for example. In the second, the IBM 7171 has recognized a problem and has disconnected the line itself. For example one of the RS-232-C signal lines may have gone inactive for a period exceeding the timeout limit. The first is called "host initiated" and the second is called "unsolicited." While either class is acceptable for disconnection, there are differences in the way in which the IBM 7171 handles them. If the disconnection is unsolicited, one cause could be trouble in the telephone system. If a new connection is then established, the IBM 7171 must make sure that some new user does not dial into the middle of the previous user's session.

5.10.1 Method of Disconnection

The IBM 7171 disconnects a port by driving DTR and RTS inactive on that port. The modem should respond by driving its lines inactive. Normally these signals are held inactive for about four seconds, but in certain situations the disconnection sequence may be repeated.

5.10.2 Types of Disconnection

Unsolicited

- Loss of DSR beyond limits.

  The IBM 7171 monitors the DSR signal from the modem. If and when DSR is driven inactive, a timer starts on that port. If DSR is not yet active when the timer runs out, the IBM 7171 will disconnect that port. That timer is set to 20 milliseconds during the initialization phase and defaults to 100 milliseconds in the operational phase. The operator has the ability to change operational phase value if desired. Use the Maintenance Facility command:

  ```
  sw DCOO:000A XXXX
  ```

  where XXXX is the new delay value expressed in hexadecimal (e.g. for a new delay value of 600 milliseconds XXXX would be 0258).

  Some modems drive DSR with a mechanical relay. Since relays give erratic signals as they switch, the IBM 7171 will allow 50 milliseconds during initialization for the bouncing to stop. A typical relay will bounce for a couple of milliseconds.

  If the modem is powered off, or if the cable becomes loose, the loss of DSR will cause the unsolicited disconnection.

- Loss of RLSD beyond limits.

  The IBM 7171 monitors the RLSD signal from the modem. This signal is sometimes called DCD (Data Carrier Detect) and is active when the modem attached to the IBM 7171 port is detecting the carrier signal from the other modem. If the carrier is lost, RLSD indicates that the other modem has been turned off, the cable is disconnected, or some similar problem. If the IBM 7171 detects that the modem has driven RLSD inactive, it will start a timer. If RLSD is not driven active again before the timer runs out, the IBM 7171 will disconnect that port. The IBM 7171 is delivered with the RLSD timer set to 500 milliseconds for all ports, but the operator can change it if desired. Use the Maintenance Facility command:

  ```
  sw DCOO:000C XXXX
  ```

  where XXXX is the new delay value expressed in hexadecimal (e.g. for a new delay value of 600 milliseconds XXXX would be 0258).

- Loss of CTS beyond limits.

The IBM 7171 monitors the CTS signal from the modem. During the line initialization of some of the higher speed modems, several seconds may elapse while the modems condition the transmission line. CTS can also be used by the modem (or null modem attached device) to pace the data being sent to it. When the IBM 7171 detects that CTS is being driven inactive at a port, it starts a ten second timer. If CTS is not driven active again before that timer expires, the IBM 7171 will disconnect that port.

- Invalid interface control lead sequences.

  Certain unexpected control signal sequences will also cause the IBM 7171 to disconnect a port. If, for example, a port is in the operational phase with data flowing back and forth between the IBM 7171 and a terminal, and the Ring Indicator should become active indicating a new incoming call is being received, the IBM 7171 will recognize an inconsistent state on that port and will disconnect it.

- Terminal Powered Off

  If a modem or directly attached terminal is powered off, the RS-232-C signals on that device will drift to their inactive states. The IBM 7171 port will detect them and will start the timers. When the first of the timers runs out, the IBM 7171 will disconnect that port.

  Warning: When a directly attached device is connected to the IBM 7171 port with a null modem cable, it is possible to assemble that cable so that the IBM 7171 port can not detect that the terminal power has been turned off. That type of cable is not recommended, and it could lead to problems maintaining user security on the host system.

Host Initiated

The host can be commanded to disconnect one of the IBM 7171 ports. This would be used as the normal end to an operating session. Section "3.1.5 Setup Functions" on page 3-3 describes the command "Keyboard initiated line drop"

5.11 Communications Error Handling

Communication systems are prone to transmission errors, and there are certain types of checks built into the transmission scheme.

The errors that are detected by the IBM 7171 are:

Parity Error Data character received from attached device has wrong parity.

Framing Error Start and stop bits are not correctly matched. (Possible false start)

Break Interrupt The received data has gone to a space (zero) level for more than one character time. (Could be Break Key was depressed.)

Overrun Error The received data has overrun the previously received data character before it could be processed.

These errors occur in the communication from the terminal to the IBM 7171. When the IBM 7171 detects one of the above errors, it notifies the user by locking the keyboard. As the user tries to enter more characters or commands, the IBM 7171 will echo back the BEL character instead of the typed character. The user can acknowledge and clear this error state by entering the "reset key sequence" described in section "3.1.5 Setup Functions" on page 3-3.

Chapter 6. IBM 7171 I/O Interface to the Host System

It is assumed that the reader is familiar with 3270 style Channel Command Words (CCW's) and data streams. They are defined in other IBM documentation.[1]

6.1 High Level Examples

Data movement between the host and its control units has its direction references in relation to the host. This means that 'outbound data' is from host to a control unit. 'Inbound data' is coming to the host from a control unit.

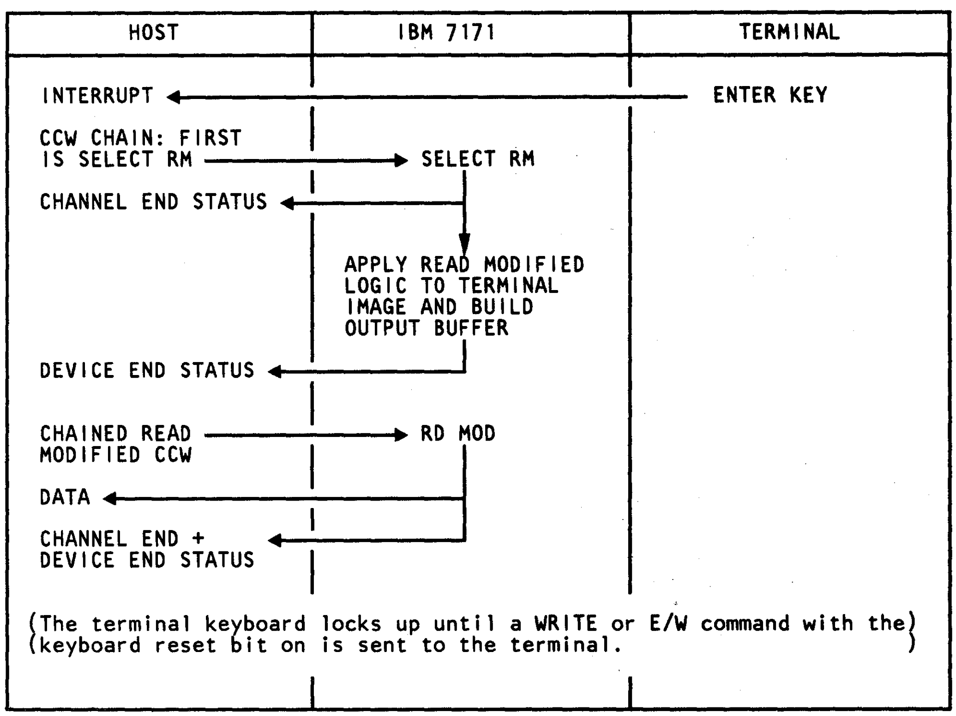


Figure 6-1. Channel Inbound Data

---

1 Refer to *IBM 3270 Information Display System: 3274 Control Unit Description and Programming Guide.*

Inbound data is signaled by an action from the terminal. In the example, in Figure 6-1, an ENTER key is hit. The IBM 7171 creates an attention interrupt for the host. The host operating system must issue some type of READ command to determine what the terminal wants. In this example a CCW chain of SELECT RM and READ MODIFIED is issued. SELECT RM is used to inform the IBM 7171 that a READ MODIFIED command will follow, and that to minimize channel busy the control unit should be waiting with the data stream. This is what the example shows, note that when the SELECT RM is received 'initial status' of Channel End is issued and the channel is freed up for the time it takes the IBM 7171 to get the data. When the data is waiting in the buffer 'ending status' of Device end is issued, informing the host that it may issue the following READ MODIFIED CCW. The READ MODIFIED command moves the data inbound immediately then supplies 'ending status' of Channel End plus Device End.

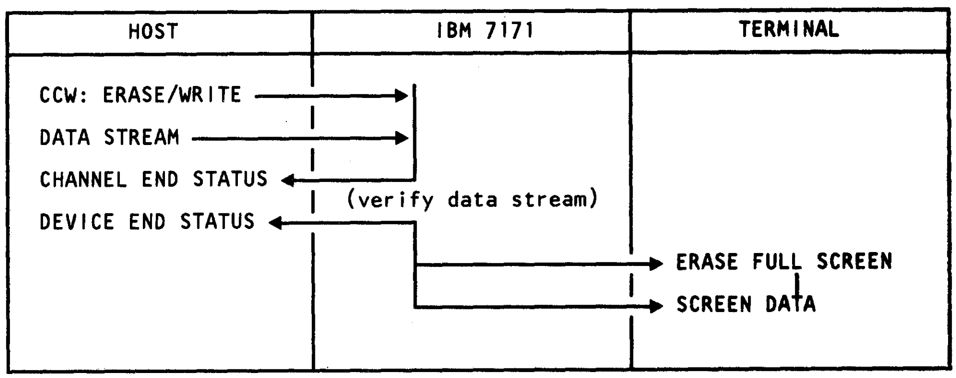


Figure 6-2. Channel Outbound Data

Outbound data is initiated by the host. In this example an ERASE/WRITE CCW command is being executed. The ERASE/WRITE command is sent to the IBM 7171, followed by the data to write on the screen. When all data is transferred to the IBM 7171, it responds with 'channel end'. Data stream verification and internal screen image updating takes place followed by DEVICE END response back to the host. The IBM 7171 now sends the new screen image to the terminal.

6.2 Data Streams

The control unit provides information to, and accepts information from, the channel at an instantaneous byte rate established by the channel or control unit, whichever is the slower.

Write command operations direct movement of data to the terminal systems via the control unit. Read command operations receive their data from the screen image that exists in the control unit.

6.2.1 Interface Codes

Data, commands, and orders transmitted between the control unit and the host system are in the form of interface codes. Two different codes are used in the United States: Extended Binary-Coded Decimal Interchange Code (EBCDIC) and American National Standard Code for Information Interchange (ASCII). The EBCDIC codes are also used in World Trade countries (ASCII is available only in the U.S.); refer to *IBM 3270 Information Display System Character Set Reference,* GA27-2837, for details. The IBM 7171 only supports the EBCDIC codes from the host.

6.2.2 Data Stream

The 3270 data stream consists of application data, commands, and orders which are transmitted between the control unit and the host system.

Data transfer commands are issued to initiate such operations as the total or partial writing, reading, and erasing of data in a selected terminal character buffer. Control commands initiate control unit operations not involved with data transfer (except for status information). Orders can be included in write data streams either alone or intermixed with display data.

Buffer control orders are interpreted and executed as they are received by the control unit, and are used to position, define, modify, assign attributes on a field and character basis, and format data being written to a display character buffer; to erase selected unprotected data in the buffer; and to reposition the cursor.

3270 Data Stream Function

The 3270 outbound data stream consists of a write control character (WCC) , and if appropriate, orders, character data, and the parameters needed by a control command. Inbound data streams consist of orders and character data or requested sense and control information.

The command defines the operation to be performed.

Commands: The operations which may be specified include:

- Write to the character buffer.
- Erase and then Write to the character buffer.
- Read the entire character buffer.
- Read only the Modified data from the character buffer (some exceptions).
- Erase All the Unprotected data from the character buffer.
- Select a terminal and initiate terminal character buffer to control unit buffer transfer of all data, modified data only, or data from position in preparation for a Read Buffer, Read Modified or a Write operation.
- Perform No functional Operation, retrieve pending status.
- Sense further definition of the Unit Check condition.
- Sense the Control Unit Identification.

Command codes are shown below.

Command	EBCDIC
Erase All Unprotected	0F
Erase/Write	05
Erase/Write Alternate	0D
No Operation	03
Read Buffer	02
Read Modified	06
Select RM	0B
Select RB	1B
Select RMP	2B
Select RBP	3B
Select WRT	4B
Sense	04
Sense ID	E4
Write	01
Test IO Instruction[2]	00

Figure 6-3. IBM 7171 CCW Command Codes

6.2.3 Write Commands

Two write-type commands, Write and Erase/Write, are used to load, format, and selectively erase device buffer data. These commands can also initiate certain device operations such as resetting the keyboard and sounding the audible alarm. Write and Erase/Write operations are identical except that Erase/Write causes complete erasure of the device buffer before the write operation is started. Thus, Erase/Write is used to load the buffer with completely new data, whereas Write can be used to modify existing buffer data.

The Erase/Write Alternate command is used by some devices to select an alternate size character buffer. The IBM 7171 currently supports this command as an Erase/Write.

Write Command

The data stream received by the IBM 7171 for a Write type command operation consists of a Write Control Character (WCC), followed by orders and/or data.

The minimum data stream for a Write type command is a one byte WCC. This is ensured because the byte count field of the write channel command word (CCW) must be set to a minimum of one or else the command code is not sent.

2 Test IO Instruction appears to the control unit as a x'00' command code, but x'00' may not be coded in a CCW.

*	RESET BIT	PRINTOUT FORMAT		START PRINT	SOUND ALARM	KYBD RESTORE	RESET MDT BITS
0	1	2	3	4	5	6	7

BITS

Figure 6-4. WCC byte format

Bit	Explanation
0,1	Used to make the WCC an EBCDIC-ASCII translatable character.[3]
2,3	Define the printout format, as follows: =00 - The NL, EM, and CR orders in the data stream determine print line length. Provides a 132 column print line when orders are not present. =01 - Specifies 40 column print line. =10 - Specifies 64 column print line. =11 - Specifies 80 column print line.
4	Start Printer bit. When set ot 1, initiates a printout operation at completion of the write operation.
5	The sound alarm bit. When set to 1, sounds the audible alarm at the selected device at the end of the operation if the device has an audible alarm.
6	Keyboard Restore bit. When set to 1, restores operation of the keyboard and resets the AID byte at the termination of the I/O command.
7	Reset MDT bits. When set to 1 all Modified Data Tag (MDT) bits in the selected device buffer are reset before any data is written or orders are executed.

Figure 6-5. Write Control Character (WCC)

This figure describes the function of each WCC bit. When the WCC specifies an operation that does not apply to the selected device (for example, if the Sound Alarm bit is set and the selected device does not have an Audible Alarm), the specified operation is ignored. When the WCC byte is followed by order or display data bytes, only the Reset MDT Bits function, if specified, is performed before the write operation; any other WCC function is deferred until all data is written and all orders are performed.

Orders and buffer data can follow the WCC character (Orders are described later in this chapter, following the "Write Commands" description). Buffer data can be written into any specified location of the buffer without erasing or modifying data in the other buffer locations. Data characters are stored in successive buffer locations until an order is encountered in the data stream which alters the buffer

---

3 Refer to *IBM 3270 Information Display System: Data Stream Programmer's Reference.*

address, or until all the data has been entered. During the write operation, the buffer address is advanced one location as each character is stored.

The buffer location where data entry starts depends upon the following considerations:

1. The starting location may be specified by an SBA order that follows the WCC. (This order is described later in this chapter under "Orders.")

2. The starting location will be the buffer address containing the cursor if the Write command is not chained.

3. The starting location will be the current buffer address if the Write command is chained.

The formatting and placement of write data and the modification of existing buffer data are described under "Orders."

Programming Notes:

1. *If the commands are being chained, the Write or Erase/Write command with the Start Print WCC bit set must be the last command in the chain. If not, the Write or Erase/Write command that specifies Start Print will be aborted.*

2. *The Printout Format bits are honored only if the Start Print bit is set in the same WCC.*

Programming Restriction: *A Write command should not be chained from an Erase All Unprotected command. If it is, the operation is undefined.*[4]

Erase/Write or Erase/Write Alternate Command

Execution of the Erase/Write or Erase/Write Alternate command performs two operations: an erase operation and a write operation. The erase operation clears the entire device buffer to nulls, positions the cursor to character location 0, and resets the buffer address to 0.

Erase/Write then performs the write and WCC operations in the same manner as a Write command.

---

[4] Refer to *IBM 3270 Information Display System: 3274 Control Unit Description and Programming Guide.*

6.2.4 Orders and Attributes

Orders

Orders can be included in Write, Erase/Write, or Erase/Write Alternate command data streams, either alone or intermixed with display or print data. Two types of orders are available: printout format orders and buffer control orders. Printout format orders are initially stored in the buffer as data and are subsequently executed only during a print operation.

The following paragraphs describe buffer control orders, which are executed as they are received in the write data stream by the IBM 7171; these orders are not stored in the buffer. Seven buffer control orders (see following figure) are provided to position, define, and format data being written into the buffer, to erase selected unprotected data in the buffer, and to reposition the cursor.

Start Field (SF) Order: This order implies that the next byte in the write data stream is an attribute character. The IBM 7171 then stores this attribute character at the current buffer address.

Note: The byte immediately following the SF order in the data stream is always stored as an attribute character, even when the byte is intended as an order or an alphameric data character.

During execution of a Read Buffer command, the IBM 7171 automatically inserts SF order codes in the read data stream immediately before each attribute character. This permits identification of the attribute characters by the program and also permits correct storage of attribute characters in the buffer if the read data is used for subsequent write operations.

Graphic Escape (GE) Order: This order implies that the next byte in the write datastream is a 3278 APL character. The IBM 7171 stores this APL character at the current buffer address.

Note: The byte immediately followin; the GE order in the data stream is always stored as a 3278 APL character (assumin; the IBM 7171 is in 3278 mode), even when the byte is intended as an order or an alphameric data character.

Set Buffer Address (SBA) Order: This three byte order specifies a new buffer address from which write operations are to start or continue. Set Buffer Address orders can be used to write data into various areas of the buffer. An SBA order can also precede another order in the data stream to specify the starting address for a PT, RA, or EUA order; to specify the address at which an attribute byte is to be stored by an SF, or to specify the address at which the cursor is to be repositioned by an IC order.

ORDER	ORDER SEQ			
	Byte 1 EBCDIC (HEX)	Byte 2	Byte 3	Byte 4
START FIELD (SF)	1D	attribute character		
GRAPHIC ESCAPE (GE)	08	Character to convert		
SET BUFFER ADDRESS (SBA)	11	1st address byte	2nd address byte	
INSERT CURSOR (IC)	13			
PROGRAM TAB (PT)	05			
REPEAT TO ADDRESS (RA)	3C	1st address byte	2nd address byte	character to be repeated
ERASE UNPROTECTED TO ADDRESS (EUA)	12	1st address byte	2nd address byte	

Figure 6-6. Buffer Control Orders and Order Codes

If the SBA order specifies an invalid address, the write operation is terminated at this point.

When a Read Modified command is executed and an attribute character (initially sent to the device by writing an SF order) is detected with the MDT bit set, the IBM 7171 inserts, in place of the attribute, an SBA code followed by the two byte buffer address of the first character in the modified field (attribute address +1). This permits identification by the control unit of fields that are modified.

Insert Cursor (IC) Order: This order repositions the cursor to the location specified by the current buffer address. Execution of this order does not change the current buffer address. For example, if IC is issued when the current buffer address is 160 and the cursor is at location 80, the cursor is moved from location 80 and inserted at location 160. The current buffer address at the end of this operation would remain 160.

Program Tab (PT) Order: The PT order advances the current buffer address to the address of the first character position of the next unprotected field. If the PT is issued when the current buffer address is the location of an attribute byte of an unprotected field, the buffer address advances to the next location of that field (one location). In addition, if the PT order in the write data stream does not follow a control command, order, or order sequence such as a WCC, IC, or RA, nulls are inserted in the buffer from the current buffer address to the end of the field, regardless of the value of bit 2 (protected/unprotected) of the attribute character for the field. When the PT order follows a control command, order, or order sequence, the buffer content is not modified for that field.

The PT order stops its search at the last location in the buffer. If an attribute character for an unprotected field is not found by this point, the buffer address is set to location 0. (If the PT order finds an attribute character for an unprotected field in the last buffer location, the buffer address is also set to zero.)

To continue the search for an unprotected field, a second PT order must be issued immediately following the first one. Since the current buffer address was reset to 0 by the first PT order, the second PT order begins its search at buffer location 0. If the previous PT order was still inserting nulls in each character location when it terminated at the last buffer location, the new PT order will continue to insert nulls from buffer location 0 to the end of the current field.

Repeat to Address (RA) Order: The RA order stores a specified alphameric or null character in all buffer locations, starting at the current buffer address and ending at (but not including) the specified stop address. This stop address and the character to be repeated are identified by the bytes immediately following the RA order in the write data stream, as follows:

BYTE	
0	RA Order
1	Stop Address (byte 1)
2	Stop Address (byte 2)
3	Character to be repeated
OR	
0	RA Order
1	Stop Address (byte 1)
2	Stop Address (byte 2)
3	Graphic Escape
4	Character to be repeated

Figure 6-7. Repeat to Address (RA) Order formats

The third character (fourth if the third is Graphic Escape) following the RA order is always interpreted as the character that will be repeated. If an invalid stop address is specified, the write operation is terminated at this point without storing the character, and error status is generated.

When the stop address is lower than the current buffer address, the RA operation wraps from the last position in the buffer to the first. When the stop address equals the current address, the specified character is stored in all buffer locations.

Attribute characters will be overwritten by the RA order if they occur before the RA order stop address.

Erase Unprotected to Address (EUA) Order: The EUA order inserts nulls in all unprotected buffer character locations, starting at the current buffer address and ending at, but not including, the specified stop address. This stop address is specified by two address bytes which immediately follow the EUA order in the write data stream. If an invalid address is specified, the write operation is terminated at this point and error status is generated.

When the stop address is lower than the current buffer address, the EUA operation wraps from the last position the buffer to the first. When the stop address equals the current address, all unprotected character locations in the buffer are erased.

Attribute characters are not affected by the EUA order.

6.2.5 Read Commands

The read-type commands executed by the IBM 7171 are Read Buffer and Read Modified. Read Buffer causes the entire buffer contents of the addressed terminal to be read into main storage. The operation initiated by Read Modified is determined by display station operator actions. The information read during execution of Read Modified could consist of fields of data modified by keyboard operations, buffer addresses, data of CURSOR SELECT fields, or the code of a Program Function or Program Access key.

An operator action that requires program interaction causes an attention interruption; the program would respond to this attention interruption with a read command.

Programming Note: *Unsolicited read commands are not recommended because the information read by these commands may be incomplete.*[5]

Read Buffer Command

Execution of the Read Buffer command causes all data in the addressed device buffer, from the buffer location at which reading starts through the last buffer location, to be transferred to main storage. This command is provided primarily for diagnostic purposes. The transfer of data begins from the current buffer address if the Read Buffer command is chained. Regardless of where the transfer of data begins, data transfer from the buffer will terminate when the last character location in the buffer has been transferred, or before the last character location has been transferred when the channel byte count reaches 0 (in this case, the buffer address after termination is undefined).

The transferred data stream begins with a three character read heading consisting of the AID character followed by a two character cursor address. The contents of all buffer locations are transferred, including nulls. Start Field (SF) orders are inserted to identify the beginning of each field. The possible Attention Identification (AID) byte configurations are shown in the following figure. An AID configuration other than 60 is set when the operator at the selected display station has performed an operation that requires program intervention. These operations are (1) pressing a Program Function or Program Access key, or (2) detecting on an attention field with the CURSOR SELECT key.

---

[5] Refer to *IBM 3270 Information Display System: 3274 Control Unit Description and Programming Guide.*

ACTION (key hit)	AID Generated (EBCDIC)	Reaction to READ MODIFIED command	Resultant data transferred to CPU
No AID generated	60	Rd Mod (unsolicited read)	Transfer field addresses and text for modified fields
No AID generated (PRINTER)	E8	Rd Mod	
ENTER key and Selector-Light pen attention	7D	Rd Mod	AID code and cursor address, followed by an SBA order, attribute address + 1 and text for each modified field. Nulls are suppressed.
PF1 key	F1	Rd Mod	
PF2 key	F2	Rd Mod	
PF3 key	F3	Rd Mod	
PF4 key	F4	Rd Mod	
PF5 key	F5	Rd Mod	
PF6 key	F6	Rd Mod	
PF7 key	F7	Rd Mod	
PF8 key	F1	Rd Mod	
PF9 key	F1	Rd Mod	
PF10 key	7A	Rd Mod	
PF11 key	7B	Rd Mod	
PF12 key	7C	Rd Mod	
PF13 key	C1	Rd Mod	
PF14 key	C2	Rd Mod	
PF15 key	C3	Rd Mod	
PF16 key	C4	Rd Mod	
PF17 key	C5	Rd Mod	
PF18 key	C6	Rd Mod	
PF19 key	C7	Rd Mod	
PF20 key	C8	Rd Mod	
PF21 key	C9	Rd Mod	
PF22 key	4A	Rd Mod	
PF23 key	4B	Rd Mod	
PF24 key	4C	Rd Mod	
Selector-Light-Pen Attention space null	7E	Rd Mod	AID code, cursor address, and field addressed only; no data.
PA1 key	6C	Short Rd	AID code only
PA2 key	6E	Short Rd	
PA3 key	6B	Short Rd	
CLEAR key	6D	Short Rd	
TEST REQ or SYSTEM REQ keys	F0	Test Req Rd	A test request message. AID transferred on Read Buffer only.

Figure 6-8. Attention ID (AID) Configurations

Read Modified Command

Read Modified initiates one of three operations, as determined by operator actions at the display station: (1) Read Modified, (2) Short Read, or (3) Test or System Request Read. The 'Attention ID (AID) Configuration' figure lists the operator actions and the resulting Read Modified command operation initiated by each action.

A major feature of Read Modified command operations is null suppression. The device buffer is cleared to all nulls when the operator turns power on or presses the CLEAR key, or when the erase portion of an Erase/Write command is executed at the selected device. Also, selected portions of a buffer can be cleared to nulls by the Erase All Unprotected command and certain orders. During Read Modified command operations, null codes are not sent.

Read Modified Operation: During a Read Modified command, if an AID other than the CURSOR SELECT key, a PA key, or the CLEAR key is generated, all fields that have been modified by a keyboard or the CURSOR SELECT key are transferred to the program. All nulls are suppressed during data transfer and thus are not included in the read data stream. As a field is modified by the operator, the modified data tag (MDT) bit is set in the attribute byte for that field. Then, when a read-modified operation is performed, successive attribute bytes are examined for a set MDT bit. When the bit is found, the data in the associated field is read (with nulls suppressed) before the next attribute byte is examined.

The first three bytes of the read data stream are always the AID code and the two byte cursor address; these bytes are called the "read heading."

Following the read heading is the alphameric data of each modified field. The data for each field is preceded in the data stream by a Set Buffer Address (SBA) order code followed by the two byte buffer address of the first character position in that field (the attribute address + 1). Thus, the read data stream when data has been modified is as follows:

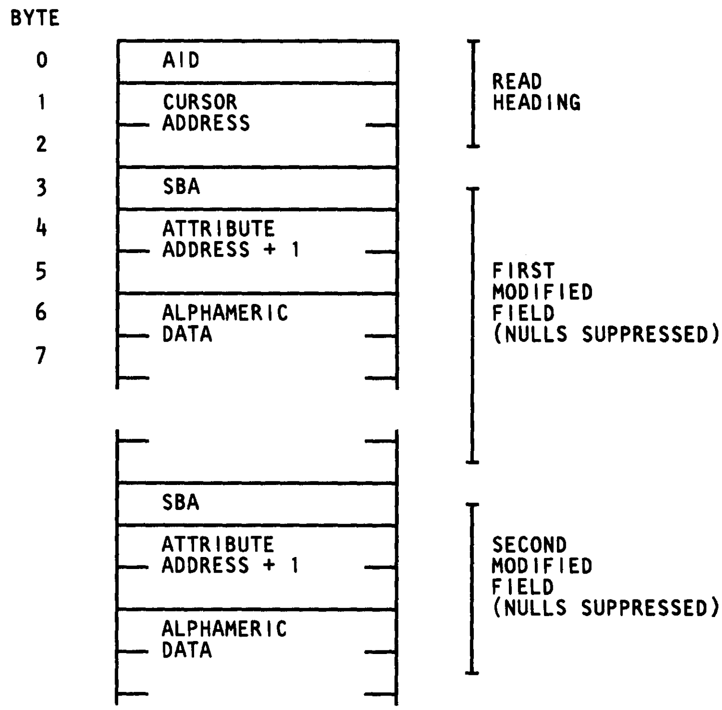


Figure 6-9. Read Modified data stream

If a space AID is generated, fields are not transferred to main storage during the read-modified operation. Instead, when a set MDT bit is found (indicating selector-light-pen and/or keyboard activity), only the Read Heading, the SBA order code, and the attribute address + 1 are transferred.

Note that if fields are modified by the keyboard but completion of the modification is signaled by CURSOR SELECT operation on other than ampersand character-designator fields, a resulting read-modified operation will read only the address of the modified fields, not the modified data.

The buffer location at which the search begins for attribute bytes that define modified fields is a function of command chaining. This location is:

1. Buffer address 0 if the Read Modified command is unchained.

2. The current address if the Read Modified command is chained.

The search for modified-field attribute bytes ends when the last buffer location is checked.

The transfer of read data is terminated as follows:

1. If the last modified field is wrapped from the last buffer location (1919) to the first location, the operation is terminated after all data in the field is transferred (nulls are suppressed). The buffer address at the end of the operation is the address of the next attribute byte in the buffer. For example, if a modified field extends from address 1900 (the attribute byte) to address 79 (wrapped field), the data from address 1901 through 79 is transferred (nulls are

suppressed); in this case, the read operation is terminated with the buffer address set to 80 (the attribute byte of the next field).

2. If the buffer does not contain a wrapped modified field, and if the channel byte count has not reached zero, the modified data stream is terminated when the last modified field is transferred; at the end of the operation, the buffer address is set to 0.

3. If the channel byte count reaches zero before all modified data is transferred, read operations are terminated and the remaining modified data is not transferred. The buffer address after termination is undefined.

If the buffer is formatted (contains fields) but none of the fields have been modified, the read data stream consists of the three byte read heading only.

If the buffer is unformatted (contains no fields), the read data stream consists of the three byte read heading followed by all alphameric data in the buffer (nulls are suppressed), even when part or all of the data has not been modified. Since an unformatted buffer contains no attribute bytes, no SBA codes with associated addresses or address characters are included in the data stream, and the modification of data cannot be determined. Data transfer starts at address 0, regardless of command chaining, and continues to the end of the buffer. At the end of the operation, the buffer address is set to 0. This read operation can also be terminated by the channel byte count reaching zero before all data is read; in this case, the buffer address after termination is undefined.

Short Read: The Read Modified command causes a short read operation if the CLEAR, or a PA key has been pressed at the selected device. During the Short Read operation, only an AID byte is transferred to main storage. This AID byte identifies the key that was pressed.

Test Request Read: The Read Modified command causes a Test Request Read operation if the test request key sequence has been pressed at the selected device. The Test Request Read data stream sent to main storage is as follows:

BYTE	
0	SOH
1	%
2	/
3	STX
4	INPUT DATA

Figure 6-10. Test Request data stream (first 4 bytes)

The remainder of the data stream is the same as described previously for Read-Modified operations, excluding the three byte read heading (AID and cursor address). If the buffer is unformatted, all alphameric data in the buffer is included in the data stream (nulls are suppressed), starting at address 0. If the buffer is formatted, each attribute byte is examined for a set MDT bit. Each time a set MDT bit is found, the alphameric data in the field associated with that bit is sent to main storage (nulls are suppressed); if no MDT bits are set, the read data stream consists of the Test Request Read heading only. The buffer location at which the search for MDT bits begins and the transfer of data ends is the same as described for Read-Modified operations.

Test Request Read function usage is determined by the access method. Normally, the operator would (1) clear the display, (2) enter test request data in a predefined format, and then (3) press the test request key sequence.

Control Commands

Control commands initiate certain control unit and/or device operations not involved with the transfer of data (other than status). Three control-type commands are executed by the IBM 7171; Select, Erase All Unprotected, and No Operation. The Select commands are all 'Prepare to Send' commands.

Select Read Modified (RM) Command: Select RM is an immediate command. The IBM 7171 executes a Select RM command by preparing for a Read-Modified operation; that is, the device buffer is searched for any modified fields and the input data stream is built. This could result in an AID only (Short Read), Test-Request-Read, or a Read-Modified data stream. If the command chained to the Select RM command is a Write command, the input data is not used. The write data stream is received and processed by the IBM 7171. If the Write command is a WCC, SBA xx only, and then chained to a Read Buffer or a Read Modified command, the input data stream that had been prepared is not used and the appropriate data stream is prepared upon receipt of the Read Buffer or Read Modified command. If the command following the Select RM is Read Buffer, the input data is not used, a Read-Buffer operation is performed, and the data is sent to the host.

The Select RM command is used to separate the control unit data preparation from the channel operation thereby decreasing channel use.

Note: The successful use of the "Prepare to Send" Select commands on the IBM 7171 requires that appropriate code be included in the access methods of the host operating system. Host operating system sysgen manuals indicate the macros that are a prerequisite.

Select Read Buffer (RB) Command: Select RB is an immediate command. The IBM 7171 executes a Select RB command by preparing for a Read-Buffer operation; that is, a Read-Buffer data stream is built. When the data stream is completed, Device End is sent to the host. If the command chained to the select RB command is not a Read Buffer, the command will not be accepted, and CE, DE, UC, OC will be sent to the host.

Select Read Modified from Position (RMP) Command: Select RMP is an immediate command. A Select RMP command is executed by recording the Read-Modified condition and returning Device End.

The commands following the Select RMP command should be a chained Write command followed by a chained Read Modified command. [The Write Command contains only four bytes (WCC, SBA xx) to set the buffer address.] If the sequence is other than as described, the command will not be accepted, and CE, DE, UC, OC will be sent to the host.

Upon receipt of the Write command, the IBM 7171 will perform the Read Modified from Position preparation, and return Device End to the host when the data stream is completed. The Read Modified command is then executed.

Select Read Buffer from Position (RBP) Command: Select RBP is an immediate command. A Select RBP command is executed by recording the Read-Buffer condition and returning Device End.

The commands following the Select RBP command should be a chained Write command followed by a chained Read Buffer command. [The Write command contains only four bytes (WCC, SBA xx) to set the buffer address.] If the sequence is other than as described, the command will not be accepted, and CE, DE, UC, OC will be sent to the host.

Upon receipt of the Write command, the IBM 7171 will perform the Read Buffer from Position preparation, and return Device End to the host when the data stream is completed. The Read Buffer command is then executed.

Select Write Command: Select WRT is an immediate command. A Select WRT command is executed by returning Device End to the host. If the chained command following the Select WRT is not a Write command, CE, DE, UC, OC will be sent to the host.

Erase All Unprotected Command: This command performs five functions at the addressed device:

1. Clears all unprotected buffer character locations to nulls.

2. Resets to 0 the MDT bit for each unprotected field.

3. Always unlocks the keyboard.

4. Resets the AID byte.

5. Repositions the cursor to the first character location in the first unprotected field of the buffer. If no unprotected fields exist, the cursor is positioned to buffer location 0.

Erase All Unprotected is an immediate type command. Upon acceptance of this command, the IBM 7171 goes "busy" and sends Channel End initial status to the channel. Upon successful completion of this command, the control unit sends Device End status asynchronously to the channel and then goes "not busy."

Programming Restriction: *Erase All Unprotected should not be chained to a Write, Erase/Write, Erase/Write Alternate, or another Erase All Unprotected command. If it is, the resulting operation is not defined.*[6]

---

6 Refer to IBM 3270 Information Display System: 3274 Control Unit Description and Programming Guide.

No Operation Command: The No Operation command performs no functional operation, but may be used to retrieve pending status. No Operation is an immediate command; therefore, Channel End and Device End normally will be presented as initial status unless pending status or a busy condition exists.

Sense Command: The Sense command should be issued in response to Unit Check status for further definition of the Unit Check condition. The IBM7171 responds to a Sense command by sending 1 byte of sense data to the channel and resets the sense register when the Device End (DE) for the command is accepted by the channel.

All other commands to the same address, except a No Operation or a Test I/O "command" (command code of X'00'), reset the sense register immediately when the command is issued. Sense commands issued to an address other than the one for which sense data is pending are responded to with a Busy and Status Modifier (B, SM) initial status indication, and the sense register is not reset. Sense should be issued following receipt of Unit Check status to ensure that valid sense information is retrieved.

CR	IR	BOC		DC	US		OC
0	1	2	3	4	5	6	7

BITS

Bit	Name	Significance
0	Command Reject (CR)	Set if the IBM 7171 has received an invalid command.
1	Intervention Required (IR)	Set if a command, other than Sense, was addressed to a device that is unavailable or is in the "not ready" condition.
2	Bus Out Check (BOC)	Set if the IBM 7171 has detected bad parity on any command or data type received from the channel.
4	Data Check (DC)	Set if: (1) the terminal keyboard typed the line drop characters, (2) the Maintenace Facility issued a FORCE DROP, (3) the RS-232-C communication link lost integrity, or (4) an inbound LOGOFF data stream was encountered.
5	Unit Specify (US)	Set if the sense bits resulted from polling the RS-232-C interface.

Figure 6-11 (Part 1 of 2). Sense Bit Description

Bit	Name	Significance
7	Operation Check (OC)	Set when the IBM 7171 has received a valid command or order that it cannot execute as follows: 1. SBA, RA, or EUA order specifies an invalid buffer address. 2. Write data stream ends before all required bytes of SBA, RA, EUA, or SF order sequences are received. 3. The IBM 7171 received a command chained to Select RB, Select RBP, Select RMP or Select WRT command other than was expected; or the byte count of a Write command after RBP or RMP was not equal to 4.

Figure 6-11 (Part 2 of 2). Sense Bit Description

Sense ID Command: The Sense ID command requests data transfer to the host. Four bytes of data are sent as follows:

```
|FF|32|74|1D|
```

Sense ID is honored when the IBM 7171 is in one of the following states:

- Power on
- IML completed
- Online
- Not busy
- No outstanding status to be presented.

6.2.6 Miscellaneous Operations

Test Request Function

The Test Request message sent to the host (SOH%/STX) is invoked from the keyboard.

Display Fields

A formatted display contains display fields defined by the program. These fields consist of blocks of character positions bounded by control characters. The control character at the start of a field is set by the program to determine the characteristics of the field; this character contains the field attributes. (For details, see "Attributes," later in this chapter.) Fields containing character positions on more than one row "wrap" from the last character position on one row to the first character position on the next row. A field may wrap the screen; if the first character position on the screen does not contain a control character, the last field on the screen wraps from the last character position to the first. (Some field-oriented operations are terminated early if the field wraps the screen; this effect is noted in the descriptions of the specific operations.)

Display fields simplify operations both for the operator and for the programmer. Headings can be displayed to prompt the operator as to the data that should be entered, and the program can identify fields that contain entered data without reading the entire display buffer. When data is being entered into a formatted display, the presence of a control character acts as a tab stop; pressing the tab key advances the cursor from its current position to the first character position in the next unprotected field. (An unprotected field is one that accepts data input from the keyboard.)

To define the start of a field, the program may issue a Write command transferring a Set Buffer Address (SBA) order and a Start Field (SF) order to the display; the specified buffer address is selected, and the control character specified by the SF order is loaded into the addressed location. Only the start of a field is defined; starting a field ends the previous field at the character position prior to the new control character.

6.2.7 Attributes

All display stations may be programmed with formatted fields. The control character at the start of each field contains the field attributes. Attributes contained in this character apply to all the data contained in the field.

Field Attributes

The field-attribute character occupies the first character position of each display field in a formatted display; the corresponding character position on the display screen is always blank. This eight bit attribute character is loaded by a Start Field order to (1) define the start of a field and (2) assign characteristics to the field. Bit positions in the character are significant; the value assigned to each bit or group of bits controls whether a specific attribute is applied.

Field Attribute Character: The following figure shows the significance of bits in the field attribute character. Characteristics set by the field attribute character are:

- Protected/Unprotected: An operator cannot enter data into or modify the content of a protected field. Input fields that require data from the operator must be unprotected.

- Alphameric/Numeric: In an unprotected input field, alphameric/numeric defines the type of data that an operator can enter into the field. This attribute has special meaning for protected fields and data entry keyboards.

  A cursor moving into a field defined with both Numeric and Protected will skip to the following Unprotected field. A cursor moving into a field defined as Alphanumeric and Protected will lock up the keyboard which requires a user reset to free.

- Nondisplay/Display/Intensified: Data contained in the field is either not displayed, displayed at normal intensity, or displayed at high intensity.

- Detectable/Nondetectable: Displayed data in a detectable field can be selected by the CURSOR SELECT key.

Field attributes are protected against input from the keyboard; however, bit 7 (Modified Data Tag) is set to 1 when the operator enters data into the field defined by the attributes. Attribute characters are not protected against operation of the CLEAR key; pressing the CLEAR key erases all locations in the display buffer.

X	X	U/P	A/N	D/SPD	RESERVED	MDT
0	1	2	3	4	5	7

BITS

EBCDIC BIT	Field Description
0,1	Value determined by contents of bits 2-7. See following figure for hexadecimal values.
2	0 = Unprotected 1 = Protected
3	0 = Alphameric 1 = Numeric *Note:* Bits 2 and 3 equal to 11 causes an automatic skip. See text.
4,5	00 = Display/not CURSOR SELECT key detectable. 01 = Display/CURSOR SELECT key detectable 10 = Intensified display/CURSOR SELECT key detectable. 11 = Non-display, non-print, non-detectable.
6	Reserved
7	Modified Data Tag (MDT): identifies modified fields during Read Modified command operations. 0 = Field has not been modified 1 = Field has been modified by the operator. Can also be set by program in data stream

Figure 6-12. Field Attribute Character Bit Assignment

Bits 2-7	EBCDIC		Bits 2-7	EBCDIC
00 0000	40		10 0000	60
00 0001	C1		10 0001	61
00 0010	C2		10 0010	E2
00 0011	C3		10 0011	E3
00 0100	C4		10 0100	E4
00 0101	C5		10 0101	E5
00 0110	C6		10 0110	E6
00 0111	C7		10 0111	E7
00 1000	C8		10 1000	E8
00 1001	C9		10 1001	E9
00 1010	4A		10 1010	6A
00 1011	4B		10 1011	6B
00 1100	4C		10 1100	6C
00 1101	4D		10 1101	6D
00 1110	4E		10 1110	6E
00 1111	4F		10 1111	6F
01 0000	50		11 0000	F0
01 0001	D1		11 0001	F1
01 0010	D2		11 0010	F2
01 0011	D3		11 0011	F3
01 0100	D4		11 0100	F4
01 0101	D5		11 0101	F5
01 0110	D6		11 0110	F6
01 0111	D7		11 0111	F7
01 1000	D8		11 1000	F8
01 1001	D9		11 1001	F9
01 1010	5A		11 1010	7A
01 1011	5B		11 1011	7B
01 1100	5C		11 1100	7C
01 1101	5D		11 1101	7D
01 1110	5E		11 1110	7E
01 1111	5F		11 1111	7F

Figure 6-13. EBCDIC values for attribute characters

6.3 IBM 7171 Printer Support

The IBM 7171 will support ASCII printers which are functionally compatible with 328x printers. The 3270 Data Stream mode for controlling printers is supported and is described below. The SNA Character String mode is not supported. For example, to run as a 328x printer on the IBM 7171, the host processor could setup this device as a 3286 model 2 printer (DEVTYPE=3286,MODEL=2) on the channel assigned to the IBM 7171.

The format control orders NL, CR, EM, and FF provide a print format function when received by a printer, as follows:

- NL (New Line). Moves the print position horizontally to the left margin and vertically down to the next line.
- CR (Carriage Return). Moves the print position horizontally to the left margin.
- EM (End of Message). Terminates the print operation.

- FF (Form Feed). Moves the print position to the top and left margin of the next page.

The WCC byte definition for printer use is shown in Figure 6-14.

Bit	Explanation
0,1	The function of these bits is to make the WCC byte an EBCDIC/ASCII-translatable graphic.
2,3	Defines printout format as follows: 00 - The NL or CR orders in data stream determine print line length, and EM designates the end of the message. Provides a 132-character print line when orders are not present. 01 - Specifies 40-character print line 10 - Specifies 64-character print line 11 - Specifies 80-character print line
4	Start-printer bit. When set to 1, initiates a printout operation at the completion of the write operation.
5	Sound-alarm bit. When set to 1, sounds the audible alarm if an audible alarm provided.
6	Keyboard-restore bit.
7	Reset MDT bit.

Figure 6-14. The WCC Byte (As Defined for Use with Printers)

In normal operation, when bits 2 and 3 of the WCC are 0, a new-line function is performed each time a valid NL character is encountered. In addition, if no valid NL is encountered before the printer reaches the end of a line (as determined by the maximum physical carriage length), the printer automatically performs NL and continues printing.

During a print operation, if line-length format is specified in bits 2 and 3 of the WCC (bits 2 and 3 not equal to 0), data characters in the printer buffer are scanned one line at a time before they are printed. A line feed is executed after each line is printed. If a line contains only null characters and one or more space characters, a line feed is performed to cause a blank line in the printout. When null characters, field attributes, or alphanumeric characters in a nonprint field are encountered, they are treated as follows.

- If in a line that contains another print field, they are printed as spaces.

- If they constitute an entire line, they are ignored, and the line feed is not performed; as a result, a blank line does not appear in the printout, and the data is compressed vertically one line.

The validity requirements for NL, CR, EM, and FF are as follows:

- NL, CR, EM, and FF are valid only when encountered in a print field during a printout that does not have a line-length format specified by the WCC.

When line-length format is not specified, printout of the buffer begins at buffer address 0 and continues until the last position of the character buffer is printed. If printer control characters are embedded in the data they are converted to the following characters:

- NL prints as a 5
- EM prints as a 9
- CR and FF print as a blank.

6.4 IBM 7171 to Host Interactions

The channel program controls all control unit operations by transmitting information across the I/O interface. This information consists of (1) an address byte, which selects one control unit and one device (display or printer) attached to the control unit; (2) command bytes, which specify the type of operation to be performed by the control unit for that device; (3) data bytes, which either are stored in the control unit buffer for ultimate use by the selected device as display or printout data or are decoded as orders and used by the control unit for formatting the buffer; and (4) various control signals. Status bytes, which are automatically generated by the control unit, inform the channel program (1) of the general condition of the control unit and selected device at various stages of command operations and (2) of unique conditions of the control unit and any attached device when command operations are not in progress.

6.4.1 Interface Operations (IBM 7171/Channel)

The host program initiates control unit operations with a Start I/O instruction. This instruction identifies the I/O control unit and device (in this case, the control unit and a display/printer) and causes the channel to fetch a channel address word (CAW) from a fixed location in main storage. The CAW designates the storage protection key and the location in main storage from which the channel subsequently fetches the first channel command word (CCW). The CCW specifies the command to be executed and the number and address, in main storage, of any bytes to be transmitted.

Up to 64 devices can attach to control units that have addresses divisible by four. Device addresses must always be assigned sequentially, starting with address 0. However, no priority is given to a particular device address.

Selection

The channel attempts to select the control unit and an attached device by sending a unique address byte to the control unit (and to all other control units attached to the same channel or subchannel). When a control unit recognizes both addresses (its own and its devices), it logically connects to the channel and responds to the selection by returning the device address byte to the channel.

Command Initiation

Command operations by the control unit start when the control unit and a device are successfully selected. When a command is to be executed by the control unit, the channel sends the command code (CCW bits 0 – 7) to the control unit.

When execution of the command involves a transfer of data (such as Write or Read Modified), the control unit responds to the command with a status byte (called "initial" status) indicating whether it can execute the command. If the command can be executed, the channel is set up to respond automatically to service requests from the control unit, and the control unit assumes further control of the operation. Command operation can be terminated by the control unit or when the channel byte count reaches 0. At this time, the control unit sends the channel a second status byte (called "ending" status) which indicates whether the command operation was successfully performed.

When the function of the 3270 command does not involve the transfer of data (such as EAU), it is called an "immediate" command. The resulting control unit operation depends on the particular command, as follows. If the command is No Operation, ending status and initial status are combined to indicate to the channel that the control unit has completed execution of the command. If the command is Select or Erase All Unprotected, which initiates certain control unit and device operations, the initial status from the control unit allows the block multiplexer channel to be released to perform other operations. When command execution is completed by the control unit (and regains selection), the control unit sends ending status to the channel, indicating whether the command was successfully performed.

Chaining

When the channel has completed the operations specified by a CCW, it can continue the activity initiated by the Start I/O by fetching a new CCW, thereby starting execution of another command. The fetching of this new CCW is called "command chaining", and the CCWs belonging to such a sequence are said to be chained. All CCWs in a chain apply to the control unit and device specified by the Start I/O instruction.

Either of two types of chaining can be specified by the current CCW (bits 32 and 33): data chaining or command chaining. During data chaining (current CCW bit 32=1), the new CCW fetched by the channel defines a new main storage area (data address) for the current command. During command chaining (current CCW bit 33=1), the new CCW specifies a new command and a data address for that new command.

Thus, when command chaining is used, the control unit is selected following the Start I/O instruction when the channel receives the first CCW in the chain that involves operations with the control unit. The control unit is dedicated to one CCW string until final Channel End time or until operations are abnormally terminated. Programming restrictions that must be observed when command chaining is used were described earlier under commands.

Status

The control unit generates a status byte to inform the channel of certain control unit and device conditions. This status byte can be generated synchronously (while the control unit is selected and performing a command operation with the channel) or asynchronously (while the control unit is not selected).

Synchronous status is passed to the channel as both both "initial" and "ending" status to a command. Initial status reflects the condition of the selected device and/or control unit upon receipt of a command, and indicates to the channel whether the command can be executed. Ending status reflects the condition of the control unit and selected device after all channel/terminal interface operations of a nonimmediate command are completed. Asynchronous status reflects (1) ending status for an immediate command other than No Operation, (2) a second ending status for a Write, Erase/Write, or Erase/Write Alternate indicating that the control-unit-to-device buffer transfer is completed, or (3) an equipment condition or operator action not associated with command execution (an attention).

Figure 6-15 "Status Byte Bit Assignments" describes each bit of the status byte. Status is reset by the control unit once it has been accepted by the channel.

Bit	Name	Condition
0	Attention (A)	Indicates a request for services from a terminal. Program should respond by issuing a Select command followed by the correct Read Modified chain. Attention bit is also set with (Unit Check) as a result of non-synchronously detected equipment malfunctions; in this case, program should respond by issuing a Sense command.
1	Status Modifier (SM)	Set with (Busy) in initial status when there is pending status for another device in this control unit.
2	Control Unit End (CUE)	Set following a Busy condition, after pending status is cleared or to indicate that the control unit is now free to accept a new command.
3	Busy (B)	Set alone in initial status byte when addressed device is already busy performing a command. Set with Status Modifier when addressed control unit is busy. When the channel addresses a device other than the one that is busy and control unit is not busy, addressed device becomes selected and the command is honored. The Busy bit is also set in pending status if addressed device has such status; if pending status is for a device other than the one addressed, Status Modifier bit is also set.

Figure 6-15 (Part 1 of 2). Status Byte Bit Assignment

Bit	Name	Condition
4	Channel End (CE)	Indicates channel data transfer operations are completed. Is set alone: 1. In initial status for Select or Erase All Unprotected command. 2. As ending status for Write, Erase/Write, and Erase/Write Alternate commands; in all cases, Device End status is sent non-synchronously when device operations (command execution or control-unit-to device-buffer transfer) are completed. Is set with Device End, to indicate that control unit and device operations are completed. 3. In initial status for No Operation command. 4. In ending status for Read Buffer, Read Modified, or Sense command. 5. Asynchronously if only Channel End status was pending and the device operation is completed before the channel accepts status. Is set with Device End and Unit Exception in initial status for Read or Write command if addressed device is busy executing another command.
5	Device End (DE)	Indicates that the control unit and device have completed all command operations and are free to execute another command. Is set: 1. In initial status for No Operation command 2. In ending status for Read Buffer, Read Modified, or Sense command. 3. In non-synchronous status for Write, Erase/Write, Erase/Write Alternate, Select, or Erase All Unprotected command.
6	Unit Check (UC)	Is set when an irregular program or equipment condition is detected by the control unit or device. Program should always respond to Unit Check status by issuing a Sense command for further definition of the error condition.

Figure 6-15 (Part 2 of 2). Status Byte Bit Assignment

The following figures list the initial, ending, and asynchronous status and sense bit combinations, respectively. The abbreviations used in these figures are as follows:

- Status Bits

  A - Attention
  B - Busy
  CE - Channel End
  DE - Device End
  SM - Status Modifier
  UE - Unit Exception
  UC - Unit Check

- Sense Bits

  BOC - Bus Out Check
  CC - Control Check
  CR - Command Reject
  DC - Data Check
  EC - Equipment Check
  IR - Intervention Required
  OC - Operation Check
  US - Unit Specify

Figure 6-16. Status/Sense Byte Bit Assignments for the IBM 7171

Initial Status: Initial status is generated by the control unit in response to initial selection by the channel. During the initial selection sequence, the status byte is sent to the channel after the control unit receives a command.

Status (Hex)	Sense (Hex)	Error Recovery Procedure	Condition
All Zeros (00)			Normal status for any command other than No Operation, Select, or Erase all Unprotected.
CE (08)			Normal status for a Select or Erase All Unprotected command.
CE,DE (0C)			Normal status for a No Operation command.
UC (02)	BOC (20)	1	A parity check was detected on the command byte.
UC (02)	IR (40)	2	A command other than Sense was addressed to a device that the control unit has recorded as "unavailable" or "not ready."

Figure 6-17 (Part 1 of 2). Initial Status and Sense Conditions

Status (Hex)	Sense (Hex)	Error Recovery Procedure	Condition
UC (02)	CR (80)	3	An invalid command was issued to the control unit.
B (10)			Response to a command addressed to a device which is being serviced by the control unit or which is completing a previously issued command.
B,SM (50)			Response to a command addressed to a device other than device whose status is pending or device being serviced by the control unit.

Figure 6-17 (Part 2 of 2). Initial Status and Sense Conditions

This figure shows the possible initial status bit configurations. An all-zero status byte is sent when a nonimmediate command is accepted for execution by the control unit; it is also sent in response to Test I/O if other status is not pending. The Unit Check bit is set if the command is not accepted by the control unit because of a program or equipment error.

Initial status to immediate commands is as follows. For No Operation, Channel End and Device End are both set to indicate completion of the command. For Select and Erase All Unprotected, which do not involve data transfer between the channel and the control unit, Channel End is set. This frees the block multiplexer channel for other operations while the command is being executed. When command execution is completed, ending status is presented asynchronously.

If a Start I/O Fast Release (SIOF) is executed by the channel, then unchained initial status becomes ending status.

When status is pending (a previous status byte is awaiting transfer to the channel), the pending status byte, with the Busy bit set, is sent to the channel in response to any command (not to a Test I/O instruction), and that command is not accepted by the control unit. For Test I/O, the pending status byte is presented without the Busy bit set. If the pending status is for a device other than the one selected during the initial command sequence, only Busy, Status Modifier (B, SM) is presented to the channel and the pending status is retained by the control unit.

Ending Status: When the control unit completes channel operations for a nonimmediate command, it sends an ending status byte to the channel, freeing the channel for other operations. This status byte always relates to the command operation that has been executed. The normal ending status byte for a Read Buffer, Read Modified, or Sense command will have only the Channel End and Device End bits set, indicating that the command has been executed. Normal ending status for a Write, Erase/Write, or Erase/Write Alternate command is Channel End alone. When the control unit to device buffer transfer is completed, ending the command operation, Device End status is sent to the channel as asynchronous status. Any error condition associated with the operation just executed will cause additional status bits to be set. Ending status causes an I/O interruption unless chaining is specified.

When the control unit has pending status, it attempts to gain selection of the channel asynchronously to pass this status. It is passed to the channel either when selection is accomplished or as initial status for the next command (with the Busy bit set), whichever occurs first.

Status (Hex)	**Sense (Hex)**	**Error Recovery Procedure**	**Condition**
CE (08)			Set at end of data stream on Write, Erase/Write or Erase/Write Alternate command.
CE,DE (OC)			Set at end of data stream on a Read Buffer, Read Modified, or Sense command or when channel byte count goes to zero on a Read Modified or Read Buffer command.
CE,DE,UC (0E)	BOC (20)	4	The control unit detected a parity error on a character in data stream of a Write, Erase/Write, or Erase/Write Alternate command.
CE,DE,UC (0E)	DC,US (0C)	1	The IBM 7171 disabled the device because of an error. (UC, IR is reported on the retry since the device requires a Power On Reset to be re-enabled).

Figure 6-18. Ending Status and Sense Conditions

Asynchronous Status: Asynchronous status reflects: (1) the ending status of an "immediate" command other than No Operation; (2) the second ending status for a Write, Erase/Write, or Erase/Write Alternate command, indicating that all command-initiated operations are completed; (3) an action by the device operator that requires program intervention (attention status); or (4) a control unit or attached device equipment malfunction.

When an asynchronous status condition occurs, the control unit attempts to gain selection by the channel, and passes this status to the channel when selection is accomplished. This status is called "pending" status until selection is accomplished. If the channel issues a command before retrieving this pending status, the pending status is returned, with the Busy bit set, in place of initial status for the command; in this case, the command is not executed, unless it is a Test I/O instruction.

When an asynchronous condition occurs at a device while the control unit is performing command operations with another device, the asynchronous status remains pending until the control unit completes the current command operation, returns ending status to the channel, and becomes not busy. The control unit then retrieves the pending status from the device and attempts to present it to the channel in the same manner as other asynchronous status.

Some other conditions of multiple status that can occur are not covered here. These conditions can be caused by multiple error conditions occurring simultaneously.

Status (Hex)	Sense (Hex)	Error Recovery Procedure	Condition
A (80)			An attention-generating action (e.g., program access key has been depressed) was performed by the operator.
DE (04)			The control unit to device buffer transfer is completed on a Write, Erase/Write, or Erase/Write Alternate command. The device becomes 'not busy' after completing an Erase All Unprotected command. The device to control unit buffer transfer is completed on a Select command. A device changes from "not available" to "available" or from "not ready" to "ready."
DE,UC (06)	OC (01)	3	A Write, Erase/Write, or Erase Write Alternate command containing a WCC with bit 4 = 1 is chained to a subsequent command. The IBM 7171 received an invalid buffer address in the data stream of a write type command, or the data stream ended before providing all characters required for an SBA, RA, SF, GE, or EUA order on a write type command. A portion of the device buffer may have been changed.
CUE (20)			The control unit had been addressed while busy, but is now not busy and is free to accept a new command.

Figure 6-19. Non-synchronous Status and Sense Conditions

Inbound Transmissions

Terminals are put in 'keyboard lock' when an AID generating key, such as an Enter, is hit. This can be reset by a Write, Erase/Write, or Erase/Write Alternate with the WCC keyboard reset bit on, or by the terminal operator sending a reset sequence. During this 'keyboard lock' time the screen can not be modified and multiple Reads will get the same data. This is in case there was an error in transmission. IBM System Control Programs such as VM/SP and MVS are coded accordingly and always issue one of the writes with 'keyboard reset' after a read. As an example, terminal users know there is a system problem when their terminals stay in 'keyboard lock'; this happens because the host didn't issue a keyboard resetting write. The terminal operator may also reset by sending a reset sequence.

Inbound transmissions result from an operator "enter" action, a host-initiated (unsolicited) read request, or a host retry of an inbound transmission.

An operator "enter" action is one that causes an attention identifier (AID) to be transmitted inbound. The host program responds with a read request. The host program must acknowledge the inbound transmission before a new inbound operation can be performed. (See "Host Acknowledgments" following.)

A host-initiated read operation is an inbound transmission not caused by an operator "enter" action. No host acknowledgment is required before a new inbound transmission can occur.

Host retry is a retransmission of the last unacknowledged inbound transmission from the device. The host must acknowledge reception of an inbound transmission before a "new" inbound transmission can take place. A host retry transmission does not cause read state transitions (read states are discussed following) and is not considered a "new" inbound transmission requiring host acknowledgment. Host retry occurs until a host acknowledgment takes place.

The type of inbound transmission is data from the device buffer (for example, modified fields of the display image). An inbound operation device characteristic (INOP), set by the controller, defines the type.

Inbound Operation Device Characteristic (INOP): The device characteristic INOP determines the operation to be performed when data is transmitted inbound on a retry transmission, or when the device is in a data pending state.

INOP is set by any of the following:

- An operator "enter" action sets INOP to Read Modified.

- Host acknowledgment of an inbound transmission sets INOP to Read Modified.

Read States: While powered on, a device is in one of three states with respect to read operations. The three states are: Normal, Data Pending, and Retry. The events that cause transitions between the states are shown in Figure 6-20.

Normal Read State. A device is in Normal read state when powered on, or prior to initiation of a new read operation, or after use of the Reset key in certain instances (see Figure 6-20).

When in Normal read state, an operator "enter" action causes the device to prepare to generate the inbound data stream and to go into a Data Pending state.

A host-initiated read operation causes the data to be transmitted with no state transitions occurring. The device remains in Normal read state.

Data Pending State.

- Data Pending Enter: the device state after an operator "enter" action occurred.

A read command received while the device is in a data pending state causes the data to be transmitted and the device to be placed in the Retry state. (See Figure 6-20.)

Retry State.

- Retry Enter: the device state after "enter" data was transmitted to the host.

While in a retry state, the last inbound transmission can be retried by means of a Read Modified command.

A host acknowledgment causes the device to revert from a retry state to the normal read state.

Events	Read States		
	Normal (1)	Data Pending (2)	Retry (3)
'Enter' Action	2	R	R
Read Command	1	3	3
Host acknowledgement	—	1	1
Reset key	—	1	1

R = Reject, no state transition.
— = No action, no state transition.

Figure 6-20. Read State Transitions

The typical scenario would be to be in 'normal' state and get an "Enter" action (user hit enter at terminal), the terminal would switch to 'Data Pending' state. Next, a 'Read' command would switch the terminal to 'Retry' state. Next, a Write command with a WCC with the 'reset keyboard' bit on would be processed, putting the terminal back into 'Normal' state.

Host Acknowledgments: After inbound transmissions resulting from operator "enter" actions, the transmission must be acknowledged before a new inbound operation can be performed.

For inbound transmissions generated by operator "enter" actions, the following are host acknowledgments:

- A Write, Erase/Write or Erase/Write Alternate command containing a WCC with the keyboard restore bit set to 1, or an Erase All Unprotected command.

- Any write transmission when the device is in Data Pending Enter read state.

Host acknowledgment resets INOP to Read Modified.

Processing of Read Commands: Read commands (Read Modified, Read Buffer) are processed as follows.

1. If the device is in Normal read state, then:

   a. Data is transmitted inbound as defined by:

      (1) The command (Read Modified, Read Buffer)

      (2) The AID (Read Modified command only).

   The device remains in Normal read state.

2. If the device is in a Data Pending state then:

   a. If the command is Read Modified and INOP specifies Read Modified, then data is transmitted as defined by:

      (1) The Read Modified command

      (2) The AID.

   b. If the command is Read Buffer, then data is transmitted as defined by:

      (1) The command.

   For items a and b, the device is placed Retry state.

3. If the device is in a Retry state, then a "retry" is performed as follows:

   a. If the command is Read Modified and INOP specifies Read Modified, then data is transmitted as defined by:

      (1) The Read Modified command

      (2) The AID.

   b. If the command is Read Buffer, then data is transmitted as defined by:

      (1) The command.

   For items a and b, the device remains in the retry state.

Keyboard Action or Data Stream states	INOP
Clear key	R
System Request key	R
TEST key 'ENTER'	R
TEST key 'EXIT'	R
Write or E/W with WCC bit 1 on	R
Power on	R
R - Reset NC - No Change	

Figure 6-21. Reset Matrix

6.4.2 Error-Recovery Procedures

IBM 7171 Device Detected Errors

Error conditions detected by the IBM 7171 or by an attached device are indicated to the program by Unit Check status. The program must respond to this status by using a Sense command for further definition of the condition. If a Sense command is not performed and the sense conditions still exist, the control unit will not honor any other interrupts from the device. Subsequent recovery operations are then determined by the combined configurations of Unit Check status bits and associated sense bits.

Referenced Error-Recovery Procedures

The recovery procedures referenced in the Error Recovery Procedure column of the preceding 'status' figures is:

1. Reconstruct the entire buffer image and retry the failing chain of commands. The sequence of commands used to reconstruct this image should start with an Erase/Write command. If, after two retries, the problem is not corrected, follow procedure 4.

2. The error indicates the device is "unavailable." Request and wait for operator intervention to "ready" the device; then, upon receipt of Device End status, retry the chain of commands.

3. A nonrecoverable program error has occurred. Examine the data stream to locate the problem.

4. Retry the failing chain of commands. If, after two retries, the problem is not corrected, follow procedure 1. A Write command can be retried if new fields have not been created in the buffer portion which has been cleared by a Program Tab or Erase Unprotected to Address order.

5. Request maintenance for the device that is giving trouble. After the repair, reconstruct the buffer image, starting with an Erase/Write command.

6.4.3 Special Order Strings

Two special illegal SBA sequences provide the Hardcopy/Transparent interfaces. In addition, special order strings beginning with X'2B5B' allow definition of most of the extended functions supported by the IBM 7171.

- **Programming an ASCII device transparently from the host** - Some ASCII hardcopy and graphic devices cannot easily simulate a 3270 screen, nor is it desirable for them to do so. It is possible to program them from the host through a special data stream.

- **Using Setup Functions** - The extended functions provided by the IBM 7171 are normally initiated by keystrokes entered at the terminal. However, it is sometimes desirable to have the host application program set up most of these extended functions via programming. "Setup Functions for Host Application Programs" on page 6-37 gives details on how to do this.

The Hardcopy/Transparent String

A Hardcopy/Transparent String request introducer sequence at the beginning of a datastream causes the remaining data to be transmitted to the ASCII device "as is." While the WCC and the six characters of the introducer conform to the normal structure of 3270 orders, the rest of the data is treated as 7-bit ASCII. The high order bit is ignored. It is recommended that the high-order bit be turned on for every character to "relocate" everything into the range X'80'-X'FF', which is treated as ordinary graphic characters by the TP access methods. In VM/SP the fullscreen Diagnose interface should be used to transmit the data.

Write: The Hardcopy/Transparent introducer consists of two SBA sequences:

```
X'115D7F110000'
```

When encountered by the IBM 7171, the rest of the buffer is flagged as a Hardcopy/Transparent request. Since no data editing is performed, any required control characters must be supplied in the data stream, including carriage return, line feed, or padding. The IBM 7171 will still perform pacing character (eg. XON/XOFF) processing during the transmission, provided pacing is enabled. Therefore, padding characters need not be coded for terminals which use a pacing character protocol. While in transparent mode, the IBM 7171 still receives characters from the terminal. The special reset characters are processed immediately and are a means of escape from a host application that has failed. These are the only keys which can alter the absolute host control over the output data. If the logical keyboard lock state applies, other characters received from the terminal are not processed and remain in the typeahead buffer. If the keyboard is logically unlocked, characters are examined as received and discarded. While in transparent mode, no ordinary characters or edit requests from the keyboard are accepted. However, attention generating keys will be parsed and their AID value remembered.

When all this data has been processed, the IBM 7171 will generate an attention interrupt to the host for this device address. This signals the host that another buffer can be transmitted to the device. If the host application issues another write to the device before receiving the attention interrupt, the previous data may be overwritten with unpredictable results.

If the host responds to this attention interrupt with a Read Buffer or Read Modified command, the IBM 7171 will respond with the null AID for 328x printers (X'E8'), followed by the current cursor address pointing to the byte following the last data sent. If the ASCII device has a keyboard, and the keyboard is unlocked by the WCC beginning the write, then it is possible to change this AID value by pressing any of the attention generating keys (ENTER, PFK, or PA). All other input will be ignored.

After signalling the end of data transmission, the IBM 7171 remains in transparent mode until the next host output is received. This insures that spurious characters are not introduced in the output data stream between two consecutive buffers of transparent output data. However, this also means that input is parsed and discarded, and while attention generating keys will be recognized, the only action will be to save their AID for the response to the next buffer of transparent data. No second attention interrupt will be generated. It is also not possible to generate a beep in this state, so input errors like line noise and buffer overrun are not signalled at this time. If the errors have not been reset in the interim, the beep will sound when transparent mode ends.

If the host application wishes to send another buffer of transparent data after having received the attention acknowledging the receipt of the previous buffer, it simply builds a new buffer beginning with a WCC and the transparent mode introducer sequence and issues another output request. While it is acceptable for the initial transparent write to be an Erase/Write operation, all subsequent transparent output mode requests must be simple Write operations (they may not be Erase/Write or Erase/Write Alternate requests).

TSO generally requires the output to end with an Insert Cursor order. If an application program does not supply it, TSO may transmit one itself, disrupting the transparent protocol. Therefore, if the last character in the data stream is an Insert Cursor order, it is logically removed from the data count and is not transmitted to the device.

Transparent mode is ended by either a Master Reset sequence received from the keyboard or a non-transparent write from the host. A non-transparent write is one which contains more than the WCC and does not begin with the transparent mode introducer sequence. It is recommended that an application program issue a non-transparent write at the end before returning to the normal TSO or CMS environment. A null WCC followed by an SBA to the upper left corner (X'40114040') will serve nicely to end transparent mode.

Write/Read: A second form of Hardcopy/Transparent String is the "Transparent Write/Read," which may be specified with the following SBA sequences:

```
X'115D7F110001'
X'115D7F110007' (READ W/ECHO)
```

When this form is used, the IBM 7171 will enter transparent mode and transmit the data to the device as in normal transparent mode. When all data has been transmitted to the device, all characters sent by the device in response will be received and saved. This receive is terminated by the receipt of an ASCII Carriage Return, or either the Master Reset or Pacing Stop characters as defined in the active terminal definition table, or by receipt of more than a screen-sized buffer of data.

Upon presentation of the automatic attention interrupt, the host must perform a Read Modified command. Data will be returned in Read Buffer-like format. A null AID X'E8' will be followed by a cursor address pointing to the byte following the last received data, followed by the data in seven-bit ASCII, with the X'80' bit set on in each byte. All input will be ignored during the transmission of the write data, and all characters received during the read (including erroneous characters and the terminating character) will be presented to the host. After the termination character is received, the terminal is left in transparent mode. Transparent Write/Read is intended for PC file transfer protocols or for limited interactive functions such as reading crosshair positions on a graphics device.

Setup Functions for Host Application Programs

The extended functions are mostly setup functions that can also be initiated from a host application program using a special order. Special orders are defined using the convention X'2B' followed by a dollar sign ("$" or X'5B') followed by one or more characters. Additional setup functions defined specifically for use by host application programs are described below. A list of the special orders available to a host application program is shown in Figure 6-22.

2B5B81	APL Mode On
2B5BC1	APL Mode Off
2B5B95	3270 Null Processing
2B5BD5	Improved Null Processing
2B5BA9	Zones Mode On
2B5BE9	Zones Mode Off
2B5B85	Reverse Enter/New Line Keys
2B5BC5	Restore Enter/New Line Keys
2B5B83	Reverse Column and Field Tab Keys
2B5BC3	Restore Column and Field Tab Keys
2B5B97	Restore Pacing
2B5BD7	Suppress Pacing
2B5BA5	Alpha in Numeric-Only Field
2B5BE5	3270 Numeric-only Fields
2B5B89	3278 Insert Mode (cancel insert across attention)
2B5BC9	3277 Insert Mode (maintain insert across attention)
2B5B84	Alternate Display of Attributes
2B5BC4	Primary Display of Attributes
2B5BE3	Set Home Line, Set Left Margin, Set Column Tabs
2B5BB0xx	Set Mark 'xx' at Current Buffer Address
2B5BB4	Clear Mark
2B5BA2	Enable Asynchronous Status Change Notification
2B5BE2	Disable Asynchronous Status Change Notification
2B5BBC	Request Type 1 Status Message
2B5BBD	Reserved
2B5BBE	Request Type 1 Status Message with Attention
2B5BBF	Reserved
2B5B4B	Host-Initiated Line Drop

Figure 6-22. List of Special Orders for Extended Functions

The setup functions in Figure 6-22 which are not described below have already been described in section "3.1.5 Setup Functions" on page 3-3. The setup functions "set column tab," "set left margin," and "set home line," which can be executed by pressing the predefined key sequence on the ASCII terminal keyboard have no individual special orders assigned. Instead, a complex string is used to define these setup functions as described under "Set Home Line, Set Left Margin, Set Column Tabs" on page 6-38.

Most of these options provide convenience features for users. For example, there may be a difference between the preferred keyboard assignment for different phases of an application. Consider a text retrieval and text entry application where the terminal user first enters a search criterion into a formatted data entry menu screen with usual 3270 fields, then is presented with part of a text document which can be edited in "power typing" mode with almost no field boundary definitions at all. Most keyboards have a big TAB and RETURN key and possibly nothing else suitable for frequent usage. These order strings would allow the TAB key to be Field Tab on the formatted menu entry screen and Column Tab on the power typing text entry screen. Similarly, RETURN could be New Line on the formatted screen and ENTER on the text screen, or vice versa. By default everything (except improved null processing) starts out like a normal 3278 and the keys do not shift around except when the program or terminal user requests them to.

Set Home Line, Set Left Margin, Set Column Tabs: There is one complex string beginning X'2B5BE3'. These introducer characters are then followed by up to 30 additional bytes of parameter data:

```
Char#    Function
 1       Home line number
 2       Left margin column
 3-30    Column tab stops
```

Each byte is formed by the binary line or column number added to the X'40' blank character. Thus the sequence

```
2B5BE34149494F
```

would set home (X'41') to the second row (first unprotected character position), the new line left margin (X'49') to column 10, and tab stops at columns 10 and 16 (X'494F'). If the write buffer does not extend to include a full 28 tab stops, only the tab stops specified will be set. The host string fully replaces all previous tab stop settings. There is no way to merge new tab stops with existing tab stops under host control. If any value in this string is out of range or out of order the rest of the string is ignored, but all values set up to that point are preserved.

Set Mark: Using the special order string "Set Mark of 'xx' at Current Buffer Address," the host application can "overlay" a single character at the current buffer address on the screen with the "mark" character specified after the X'B0' order. The mark is invisible to the host; a host read will only receive the actual "underlying" character.

Clear Mark: With the special order string "Clear Mark," the host application can clear the mark previously set by the "Set Mark" order. After the mark has been cleared, the original contents of the screen position reappears.

Multiple Setup Format: It is possible to include several setup "orders" following a single x'2B5B' sequence by bracketing them with "$" (x'5B') characters. It is important in this case to code each order string at its full length, since x'5B' could also occur as valid data within an order string. In other words, all 28 tabs must be included (even if they are x'40'; see below).

For example, the following extended setup order might occur imbedded in a write buffer:

```
Hex  . . . 2B5B5BC5B05BE34149494F5B4040. . .405B . . .
Char.        $ $ E   $ T         $          $
Labels     E E B R M   T H N t t t t t . . .t B
```

The sequence begins with the special escape sequence (labeled E), followed by the first of the two bracketing characters (labeled B), then an order to restore Enter/New Line (labeled R). Next is a Set Mark order (labeled M) followed by the mark character ("$"), and finally, an order to establish Home (H), New Line (N) and column tabs (t t t . . .). Note that there must be exactly 28 tabs (since this is a bracketed sequence) and that "$" may be one of them (meaning column 27).

Terminal Status Reporting: The host may request certain status information about the terminal from the IBM 7171. If a status request order string is included anywhere in a buffer written to the terminal, the IBM 7171 will be primed to respond to the next host Read Modified command with a status message. This message begins with a special AID byte (X'E4' for status 1, X'E5' reserved), followed by the cursor address as usual. The remainder of the message is returned as follows:

- Status 1 - begins with an SBA byte, followed by the address of the "mark," if any, or screen address zero (x'4040'). Then, as contents of the "field," a multiple setup format string describing the current state of all host settable options is returned. The format of this string is such that, if appended to a WCC and sent to the terminal, it will re-establish the current status with respect to all reported modes. For example, the following string might be returned:

  ```
  Hex       11C1502B5B5BB06CD5A9C583A2E597C189C4
  Char.             $ $   % N z E c s V p A i D
  Label     SBA                          cont. below

  Hex       E3424A454A4F54595E63686D72777C4040...405B'
  Char.     T                                     $
  Label       (home,new line, & 28 column tabs)
  ```

  The string will always be 49 or 50 characters long, depending on whether "mark" is turned on, and will include all 28 tabs, with trailing x'40' characters for undefined entries.

- Status 2 - Reserved

Status may be requested with or without attention. Thus the host program may specify that the IBM 7171 is to generate an attention immediately after receiving the buffer containing the status request. The host may then respond with a Read Modified, and read the status response. Alternately, the host may perform the Read Modified chained to or immediately following the write containing the status request, without requesting or waiting for an attention.

Note: Status should not be requested while the keyboard is logically unlocked, as keyboard-generated attentions and AID bytes may be confused with status reports. If the attention generating form of status request were used while the keyboard were unlocked, attentions would occasionally be lost.

Asynchronous Status Change Notification: The host program may receive notification of changes in terminal status caused by the keyboard operator. When Asynchronous Status Change Notification is enabled, any keyboard initiated change in status will cause the two high-order bits of the first address byte in the cursor address field of the next Read Modified buffer sent to the host to be set to B'10......' (a value which is formally illegal and reserved, but which is seldom checked). The host application may then request status and observe changes since an earlier request.

Host-Initiated Line Drop: The special order string X'2B5B4B' occurring anywhere in an output buffer will cause the IBM 7171 to drop the communication line to the terminal after the entire buffer has been read out. This will have the effect of hanging up the phone on a dial-up connection. The IBM 7171 then re-enables the line to permit another user to dial up.

Chapter 7. Diagnostic Program

7.1 IBM 7171 Diagnostics

This section describes IBM 7171 Resident Diagnostics which are run when the IBM 7171 power is turned on. The power-on sequence automatically starts the execution of the Basic Assurance Diagnostic Tests (BAT) contained in ROMs located on the CC board, the CPU board and each TC board. Successful completion of the Diagnostic Tests results in the system becoming operational and the IBM 7171 Ready light being turned on. Error conditions are indicated by lights on the TC boards, the CPU board, and the CC board.

7.2 System Description

Figure 7-1 below shows the manner of invoking diagnostics.

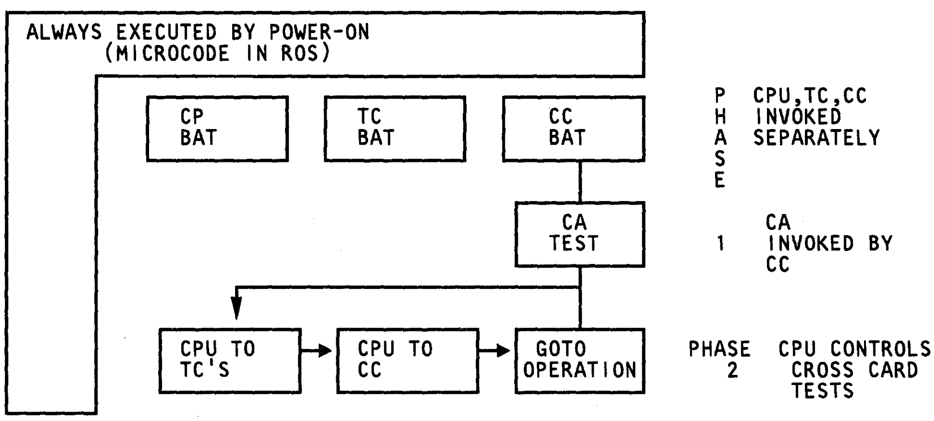


Figure 7-1. Power-on Basic Assurance Test

Basic assurance tests are run in two phases. In the first phase tests are conducted within each board containing a microprocessor (the CC board,the CPU board and the TC boards). These tests cover hardware internal to each Customer Replaceable Unit (CRU) containing a microprocessor except for the CC which also tests the CA board. In the second phase, cross board tests are under control of the CPU board which checks the interface between it and the TC boards and between it and the CC board.

After the CPU diagnostic microcode has verified the successful completion of all cross board tests it will transfer control to the operational microcode and turn on the IBM 7171 Ready light.

7.3 Error Conditions

If the Ready indicator does not light, it is an indication to the customer that a malfunction exists in the IBM 7171. In most instances, the failure can be found by the operator using the Problem Determination Procedure (PDP) which is described in the *IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide*.

Chapter 8. Problem Determination

When a problem occurs with the IBM 7171, it is the user's responsibility to take the following actions before calling for IBM assistance:

1. First, determine if the operator console is logged on. (Chapter 9 describes how to activate and use the operator console.) If it is not, attempt to log-on. If it can not be logged on then attempt recovery procedures.
2. Next, determine if the operator console is operational. If it is, record error information and then attempt recovery procedures. If the console is not operational recovery procedures should be initiated.
3. Then, if the recovery attempt is unsuccessful, try to isolate the source of the error and call the IBM Support Number for assistance.

These steps are explained in more detail below.

8.1 Recording Error Information

Errors are often the result of some transient condition, such as a momentary power fluctuation. However, the problem may also be more serious, in which case it is important to do the following (if the operator console is operational) before attempting recovery:

1. Write down any messages exactly as they appear on the operator console.
2. Take a storage dump of the IBM 7171 using the Maintenance Facility described in Chapter 9.

It is important to record the information found at the time of the failure exactly as it appears.

Note: Using a terminal with a local print option greatly simplifies this task.

8.2 Recovery Procedure

The recovery procedure consists of the following steps:

1. Activate the ON LINE/OFF LINE switch to the OFF LINE position.

2. Depress the Power Switch ON (|).

3. If the Ready indicator does not light, go to the Customer Problem Determination Procedures (CPDP) described in Chapter 7, section 7.3, Error Conditions.

4. If the Ready indicator does light, activate the ON LINE/OFF LINE switch to the ON LINE position.

5. If the problem recurs, repeat the steps above.

6. If after three tries the problem has not been resolved, call the IBM Support Number. However, before placing a service call complete the following procedure:

   a. Attempt to isolate the area where the error probably occurred.
   b. Gather information and have it available before calling the IBM Support Number.

8.3 System Problems

The following section describes some of the most common system problems.

8.3.1 Hardware Malfunctions

Symptom Ready light off and Board lights on
Reason Diagnostic failure
Action Goto diagnostic for Customer Problem Determination Procedure (Chapter 7)

8.3.2 Host Channel SYSGEN

Symptom All terminals not able to log onto the system.
Reason The CA address switches were not set to the same addresses that where specified by the host SYSGEN.
Action Set the CA switches to the same address as the SYSGEN parameters or regenerate the host system to the same addresses on the CA board.

8.3.3 Keyboard Lock

Occasionally, a system failure becomes obvious when ENTER or a PF key is pressed. The cursor goes to the lower right "keyboard lock" position, after which nothing happens. The obvious possibilities are that the host program could be in a loop or the host system is down. However, if it is not a host problem, the cause is probably the IBM 7171 Channel Interface which is discussed below.

8.3.4 Channel Interface

Intermittent channel errors are reported to the host processor as either synchronous status (while the control unit is selected and performing a command operation with the channel) or asynchronous status (while the control unit is not selected). See Chapter 6 for different status conditions which are reported.

Permanent channel errors would manifest as satisfactory terminal operation until host intervention (e.g. PA1 KEY processing) is required. If the host is functional then the problem is in the IBM 7171 and the recovery procedure described above is required.

8.4 Terminal Problems

The following section describes some of the most common errors the user could experience at a terminal. Other errors are described in "Error Situations" in Section 3.2.

8.4.1 Cabling Errors

Incorrectly wired cables can cause a variety of errors. This section describes some of the most common errors, the reasons they might occur and what action can be taken to correct the error.

Symptom The message "ENTER TERMINAL TYPE" is periodically displayed on a terminal, or within four seconds after depressing a key.

Reason The Ring Indicator (pin 22) may be "floating" in the cable, if the line is configured for Local Attach or Leased Line. Noise from other wires in the cable may induce false Ring indications out of sequence with respect to RS-232-C. In such a case, the line is dropped and subsequently restarted.

Action Do one of the following:

- Verify that the cable is properly wired, specifically the Ring Indicator lead. (pin 22)
- If configuration mode is undefined and default mode is "auto configure," then define the configuration as Local Attach or Leased Line, via the configuration override command using the Maintenance Facility.

8.4.2 Terminal Type Specification Errors

If no device type is specified for a terminal in Non-Volatile RAM (NV-RAM), the IBM 7171 assumes that the device is a terminal, and requests the terminal type from the user. If the wrong type (one that does not match the terminal) is entered even though it is a valid terminal type, the IBM 7171 will use the wrong Terminal Definition Table. This will probably result in the terminal malfunctioning.

To correct this situation, press the "Return to ENTER TERMINAL TYPE Message:" key as described in "Return to ENTER TERMINAL TYPE Message" on page 3-9.

If the user cannot get back to the "ENTER TERMINAL TYPE:" message via entries at the keyboard, he must cause the IBM 7171 to restart the communication line. This can be done for a locally attached terminal by powering the terminal off. For remotely connected lines, intervening communications equipment (modems, port selectors, etc.) may prevent proper restart when the terminal is powered on and off.

Under no circumstances will the MASTER RESET function defined in section "3.1.6 Local Reset and Control Facilities" on page 3-10 be useful in recovering from an incorrectly chosen terminal type table.

Chapter 9. Special Maintenance Facility and System Messages

9.1 Brief Overview

The special Maintenance Facility allows added control of the IBM 7171. This facility provides visibility into the IBM 7171, as well as a level of control over the operating environment. Entry to this facility is available only to the terminal attached to port 0 of Terminal Controller (TC) 0. When functioning in this facility the attached terminal is referred to as the "operator console" and the person using that terminal as the "console operator."

The console operator has the ability to examine and store data into the IBM 7171 memory. All IBM 7171 error messages are logged and reported to the operator console. The Maintenance Facility can be used for debugging as well as monitoring functions.

This facility allows the user to maintain both his host session, and his operator console session concurrently. In addition he may toggle back and forth between his host session screen and the operator console screen. Information is not lost on either session.

9.2 How to Invoke the Maintenance Facility

This facility can only be accessed through the terminal plugged into port 0 of TC 0, and is password protected. When the terminal is connected to the IBM 7171, the message:

```
ENTER TERMINAL TYPE:
```

appears. To enter this facility, enter the maintenance password, which is initially set to:

```
OPERATOR
```

The IBM 7171 then responds again with the same message:

```
ENTER TERMINAL TYPE:
```

Now key in the terminal type being used. The host session logo should now appear. To invoke the Maintenance Facility press Ctrl-W on the keyboard. The screen should now go blank. The Maintenance Facility is now active and commands may be typed in. To toggle back to the normal host-session screen, press Ctrl-W.

If an internal IBM 7171 error occurs during a host session the screen will be automatically toggled to operator console mode and a message describing the error that occurred will be displayed. To return to the host session screen, simply press Ctrl-W again.

Warning: Although the same terminal may be toggled between operator console screens and host session screens, host sessions that use the host/7171 transparency mode should not be used since unexpected results may occur.

Note: The Ctrl-W character is defined in the reset character sequence for each terminal. It is possible to re-define the sequence that initiates the Maintenance Facility toggle. For details refer to "4.4.4 TDT Area Layout" on page 4-28.

9.3 How to Use Maintenance Facility

This facility provides a full-screen user interface. All normal IBM 7171 keyboard commands are still legal for the operator console. Each line on the screen is defined as a field. Forward field tabs, backward field tabs, insert, delete etc. are all valid. If a command is typed incorrectly, the cursor can be repositioned to the line on which the error occurred, and the mistake can be corrected. This facility interprets whatever is on the line the cursor is on at the time ENTER is pressed. Characters that are past the end of the command are ignored. One command may not exceed one line in length.

Spacing is critical to all commands. Typically, each command is one or two letters in length, followed by a space. There should be a space between each argument as well. All addresses are given as:

`seg:offs`

Where:

`seg`

and

`offs`

are hex numbers. Legal characters for hex digits are 0-9 and A-F, no distinction is made between upper and lower case.

All input precision is limited to 16 bits. Any hex input larger than X'FFFF', or any decimal input larger than 65,535 will generate an error.

Warning: This facility is a very powerful debugging tool, however, since it is so powerful it should be used with extreme caution. When executing the store command, the user is changing IBM 7171 memory and must be extremely careful. Further, the memory-mapped I/O area will be affected simply by READING. For this reason, DO NOT DISPLAY memory mapped I/O. This area is from address X'84000' to X'8428C'. Unpredictable results will occur if the user tries to display this area.

9.4 Maintenance Facility Error Handling

The IBM 7171 has a system to keep track of internal system errors as they occur. The control of the error facility is governed by the user at the operator console. The errors are recorded in Non-Volatile RAM (NV-RAM) and remain in memory. If errors occur and this facility has not been activated using the Maintenance Facility password, they are still logged. These messages will be displayed on the operator console as soon as the Maintenance Facility becomes active. Errors are reported directly to the operator console if the Maintenance Facility is active, but if the normal host session is selected at the time, the terminal is switched directly to operator console mode and the error message is displayed.

The error messages are kept in a circular list, and are numbered starting from 0 to 11. The errors are recorded sequentially as they appear, and are kept in memory until written over by another error message. Thus, after error message number 11 is logged, the next error message will over-write message 0.

9.5 How to Change the Password and Parse Tables

The Maintenance Facility gives the user a great deal of control over the IBM 7171. It is therefore password protected for extra protection. The Maintenance Facility password is stored in NV-RAM and is maintained while the IBM 7171 is powered down. This password is referred to as the Maintenance password, and is initially set to 'OPERATOR'.

The interface for an automated table generating program is also password protected. The password for this interface is known as the ZAP password, and is initially set to '@@ZAP@@'. The provided IBM 7171 Support Utility described in "Appendix C. IBM 7171 Support Utility for Modifying Terminal Tables" communicates to the IBM 7171 via the ZAP interface, and also uses the ZAP password.

The password changing function of the Maintenance Facility allows the password to be changed easily. The user is strongly advised to change both of these passwords as soon as the system is installed and as often as necessary to maintain the integrity of the system.

Note: Whenever NV-RAM is re-written, or new NV-RAM is installed, the passwords will revert to their initial settings and should be reset to the installation choice as soon as possible.

It is also possible to enter terminal tables, and other information into NV-RAM through this facility. The data is entered via the store command. For a complete description of the data in NV-RAM, refer to "Chapter 4. Customizing IBM 7171 Tables" of this document.

9.6 Console Operator Commands

The following pages contain a description of each of the console operator commands. They are:

A	Display Data Control Block (AC2)
C	Configure Ports
CM	Display CC Board Memory
CO	Copy data block
CS	Store to CC Board Memory
D	Date
E	Error Message Display
FD	Force Terminal to Drop Line
FT	Force Terminal to 'ENTER TERMINAL TYPE' message
H	Help
I	Inword
M	Display CPU Board Memory
MW	Display one Word of CPU Board Memory
N	Send Notes to Terminals
O	Outword
P	Port Configuration
PW	Password Change
S	Store to CPU Board Memory
SW	Store one Word to CPU Board Memory
T	Terminal Characteristic Display
TM	Display Terminal Controller (TC) Board Memory
TS	Store to Terminal Controller (TC) Board Memory
X	Toggle default character output (ASCII/EBCDIC)
=	Perform calculator functions

9.7 A - Display AC2 Data Block for a Device

Use the A command to display the AC2 block for a specific device, or AC2.

Format:

A	device [len]
A	addr [len]

Where:

device Is a decimal device number in the range 0-63.

addr Is the address of an AC2. This address is the segment value for the AC2. No check is done to see that the given segment is indeed an AC2.

len Is an optional length, in bytes, to be displayed. If this argument is omitted, 80 bytes is the default length.

Example:

```
a 56
```

```
a 36c
```

Both of these commands might result in the following response:

```
036C:0000  C234 5678 9ABC DEF0 C234 5678 9ABC DEF0  |B......0B......0| E
036C:0010  C234 5678 9ABC DEF0 C234 5678 9ABC DEF0  |B......0B......0| E
036C:0020  C234 5678 9ABC DEF0 C234 5678 9ABC DEF0  |B......0B......0| E
036C:0030  C234 5678 9ABC DEF0 C234 5678 9ABC DEF0  |B......0B......0| E
036C:0040  C234 5678 9ABC DEF0 C234 5678 9ABC DEF0  |B......0B......0| E
036C:0050  C234 5678 9ABC DEF0 C234 5678 9ABC DEF0  |B......0B......0| E
036C:0060  C234 5678 9ABC DEF0 C234 5678 9ABC DEF0  |B......0B......0| E
036C:0070  C234 5678 9ABC DEF0 C234 5678 9ABC DEF0  |B......0B......0| E
```

The A/E to the right of the dump indicates whether the characters displayed are the ASCII or EBCDIC translations of the data dumped.

Error Messages:

Invalid device or AC2 value given.
Device not attached.
Invalid device number.
Invalid Length Specification.

9.8 C - Configure IBM 7171 Ports

Use the C command to configure any of the ports that are attached to the IBM 7171.

Format:

C	port
C	port opt [opt ...]

Where:

port Is a decimal device number.

opt Is a one character option followed by an argument. The options can be any one of the following:

Option **Argument**

b Baud rate. The following baud rates are supported and are autobaud detectable: 300, 600, 1200, 1800, 2400, 3600, 4800, 9600 and 19200 baud. If the baud rate is set to 0, then the terminal will default to autobaud detect mode.

The following baud rates are supported, but not autobaud detectable: 50, 75, 110, 134, 150, 2000, and 7200 baud.

c Number of columns on the screen. 0 defaults to 80 columns.

f Flag word. There are two bytes of flags. The meanings of these flags are as follows:

Byte 1 Flags.

Bits ***Meaning***

10 Parity. Off or 0 = Odd Parity, On or 1 = Even Parity

08 Parity Enable. Off or 0 = Parity Disabled, On or 1 = Parity Enabled

04 Stop Bits. Off or 0 = 1 stop bit, On or 1 = 2 stop bits

02-01 Data Bits. 00 = 5 data bits, 01 = 6 data bits, 10 = 7 data bits, 11 = 8 data bits.

Byte 2 Flags.

Bits	***Meaning***
02-01	Type of connection. 00 = Let TC default, 01 = Switched Network (Telephone Line), 10 = Leased Line, 11 = Direct Connect If both bits are zero, the terminal controller (TC) will automatically determine the type of line that is attached.

r The number of rows on the screen. 0 defaults to 24 rows

t Terminal type to automatically assign port at line-connect time. This is the name of a terminal whose Terminal Definition Table is currently stored in the IBM 7171. If this field is 0, the terminal user will be prompted with the 'ENTER TERMINAL TYPE' message.

If no options are present, the current status of the port is displayed.

Notes:

1. Options may appear in any order on the command line. One or all options may be set at once. The format is free-form. However, there must be one space between options, and the argument must immediately follow the option.

2. Options are processed as they are encountered. If two identical options are found on the same command line, the latter option is the one that will take effect. If an error is encountered in one option, processing is stopped. The options listed before the error will have been changed, the ones appearing after the error will not have been changed.

3. Most changes will take effect AFTER the next line-drop. The only exception is the Connection Type field in the flag word. These changes only take effect at the next IBM 7171 power-up.

Example:

```
c 56 b9600 r10 c80 tIBM3101 f1A03 r24
```

This command configures port 56 to the following:

1. 9600 baud.

2. 24 rows. Note that there are two ROW options on the same line.

3. 80 columns.

4. This terminal is assumed to be an IBM3101 and the ENTER TERMINAL TYPE message will not be displayed.

5. The flags set the terminal as follows: Even Parity, Parity Enabled, 1 stop bit, 7 data bits, force to direct-connect.

```
c 56
```

This command requests the terminal status for port 56. a response might be:

```
Dvc #56  Baud: 9600  Rs,Cs: 80,24  Flags: 1A03  Terminal: IBM3101  AC2: 037D
```

This would be a response after the configure command listed above.

Note: If Maintenance Facility cannot determine the TERMINAL TYPE from the information stored, the actual address will be displayed.

Error Messages:

Invalid device number.
Invalid configure option encountered.
Bad row value given
Bad column value given
Bad baud rate given
Bad flag value given
Bad terminal type given

9.9 CM - Display Channel Controller (CC) Board Memory

Use the CM command to display the channel controller (CC) board memory. Up to hex FF bytes of CC memory may be displayed at one time. Values may be entered in either upper or lower case, and leading zeros are not required.

Format:

CM	seg:offs len
CM	seg:offs

Where:

seg Specifies the memory segment value, in hex.

offs Specifies the memory offset, in hex.

len Indicates the number of bytes to be displayed. If length is omitted, one byte will be displayed. If length specified is larger than hex FF, FF will be the default.

Example:

```
cm 0:1a0 22
```

This command requests a display of hex 22 bytes of CC memory starting at segment 0 and offset 1A0. A response might be:

```
c 0000:01A0  1234 5678 9ABC DEF0 1234 5678 9ABC DEF0  |.4Vx.....4Vx....| A
c 0000:01B0  1234 5678 9ABC DEF0 1234 5678 9ABC DEF0  |.4Vx.....4Vx....| A
c 0000:01C0  1234                                     |.4              | A
```

The 'c' indicates that CC memory is being displayed.

The A/E to the right of the dump indicates whether the characters displayed are the ASCII or EBCDIC translations of the data dumped.

Error Messages:

Invalid Segment Specification.
Invalid Offset Specification.
Invalid Length Specification.
Address specified wraps addressability boundary.

9.10 CO - Copy Data Block

Use the CO command to copy a block of data from a source (segment:offset) address to a destination (segment:offset) address. Up to hex 400 bytes can be copied at one time. Values may be entered in either upper or lower case, and leading zeros are not required.

Format:

CO	sseg:soffs dseg:doffs len
CO	sseg:soffs dseg:doffs

Where:

sseg Specifies the source memory segment value, in hex.

soffs Specifies the source memory offset, in hex.

dseg Specifies the destination memory segment value, in hex.

doffs Specifies the destination memory offset, in hex.

len Indicates the number of bytes to be copied. If length is omitted, one byte will be copied. If length specified is larger than hex 400, hex 400 will be the default.

Example:

```
co dc00:4ce6 dc00:800 50
```

This command copies hex 50 bytes from address starting at dc00:4ce6(source segment:offset) to address starting at dc00:800 (destination segment:offset).

Error Messages:

Invalid Segment Specification.
Invalid Offset Specification.
Invalid Length Specification.

9.11 CS - Store to CC Board Memory

Use the CS command to store to CC board memory. Only one byte at a time may be stored to CC memory. Values may be entered in either upper or lower case, and leading zeros are not required.

Format:

CS	seg:offs byte

Where:

seg Specifies the memory segment value, in hex.

offs Specifies the memory offset, in hex.

byte Is the one byte that you wish to store.

Example:

```
cs 0:1a0 b3
```

This command stores the value B3 in CC memory at segment 0 and offset 1A0. No response is given.

Error Messages:

Invalid Segment Specification.
Invalid Offset Specification.
Invalid Storage Byte Format.

9.12 D - Date Commands

Use the DATE command to set the internal IBM 7171 date constant, or to display the current IBM 7171 date constant. Only decimal numerals are allowed.

Format:

D	
D	mm-dd-yy

Where:

mm Specifies the decimal month, to be no larger than 12.

dd Specifies the day, to be no larger than 31.

yy Specifies the year, to be no larger than 99.

The command D by itself is a request to display the current date constant D followed by a new date sets the internal IBM 7171 date constant to the specification given.

Note: The date command is intended only to be used as a relative time stamp for error messages. It is not updated by the IBM 7171. The date constant is only as useful as the user makes it. It is not intended to be a calendar substitute. It is recommended that the user set the date constant as often as possible. Since error messages are kept in a circular list, it would be almost impossible to determine exactly when an error occurred if the date constant were not changed.

Example:

```
D 6-30-84
```

This command sets the IBM 7171 date constant to June 30, 1984. To verify the date set, the Maintenance Facility will respond with:

```
Current Date: 6-30-84
```

Error Messages:

Invalid Date Specification.

9.13 E - Display Error Message Information

Use the E command to display error messages. All values given are in decimal. Values must not be larger than the maximum number of errors in the list.

Format:

E	
E	m1
E	m1 m2

Where:

m1 Refers to the first error message to display.

m2 Refers to the last error message to display.

If no arguments are given, the number of the current error message is displayed.

If only m1 is specified, only that error message is displayed.

If both m1 and m2 are specified, Error messages are displayed starting at m1, and continuing through m2. The command "e 1 3" will display messages 1 thru 3. Since the list of error messages is continuous, the command "e 2 1" will display the entire list of error messages.

Example:

```
e
```

This command requests a display of the current error number. A response might be:

```
Current Error Message Number: 0
```

This informs the user that the last error message written was to error message number 0.

```
e 0
```

This is a request to display error message number 0. A response might be:

```
Msg #0 Date: 6-30-84 Dvc # 63  Parameters P0,P1,P2: 0034 F502 1234
Received Unsolicited DMA.
```

Where:

Msg # Indicates which position in the error message list is being displayed.

Date: Indicates the IBM 7171 date constant on which the error message was logged.

Device # Indicates which device number, in decimal, logged the error.

Parameters List three parameters that were saved at the time of the error. Refer to the section "9.30 System Error Messages" for the meaning of the parameters.

Message Is the description of the error that occurred.

Error Messages:

Invalid Error Numbers Given.
Error Numbers Given Are Out Of Range.

9.14 FD/FT - Force a Device Offline.

Use the FD command to drop a terminal's line. Use the FT command to force a terminal back to the ENTER TERMINAL TYPE message.

Format:

FD	dev
FT	dev

Where:

dev Specifies the device number to be forced, in decimal. Only device numbers from 0 to 63 are legal.

The FT command will force the terminal back to the 'ENTER TERMINAL TYPE' message. However, the line will remain connected. The FD command will drop the connection between the terminal and the IBM 7171. In both cases, a line disconnect is sent to the host.

Example:

```
ft 56
```

This command will force the device attached to port 56 back to the 'ENTER TERMINAL TYPE' message. To determine device number, see the PORT command.

Error Messages:

Invalid Device Number.
Device Not Attached.

9.15 H - Help Command

Use the H command to display a list of all valid commands.

Format:

H	

Example:

```
h
```

The help response is:

```
VALID COMMANDS ARE:
Ac2 display, Cc Memory, Cc Store, Configure ports, COpy data block, Date,
Error display, Force Terminal message, Force Drop line, Help, Inword,
Memory display, Memory Word, Note, Outword, display Port status, Store byte,
Store Word, Terminal data display, Tc Memory display, Tc Store byte,
Xchange ascii/ebcdic, = calculate
```

The command abbreviation is found in uppercase.

9.16 I - Inword Command

Use the I command to display a word from the specified I/O port. The port value is specified in hex. The port may be entered in either upper or lower case, and leading zeros are not required.

Format:

I	port

Where:

port Specifies the I/O port to be read. The port number is specified in hex, and the response is in hex.

Example:

```
I ffc0
```

This command requests a read of I/O port FFC0. A response might be:

```
0120
```

Error Messages:

Invalid I/O Port Number Specification.

9.17 M - Display CPU Board Memory

Use the M command to display CPU board memory. Up to hex 150 bytes of CPU memory may be displayed at one time. Values may be entered in either upper or lower case, and leading zeros are not required.

Format:

M	seg:offs len
M	seg:offs

Where:

seg Specifies the memory segment value, in hex.

offs Specifies the memory offset, in hex.

len Indicates the number of bytes (in hex) to be displayed. If length is omitted, one byte will be displayed. If the length specified is larger than hex 150, 150 will be the default.

Example:

```
m dc00:0 25
```

This command requests a display of hex 25 bytes of CPU memory starting at segment DC00 and offset 0. A response might be:

```
DC00:0000  1234 5678 9ABC DEF0 1234 5678 9ABC DEF0  |.4Vx.....4Vx....| A
DC00:0010  1234 5678 9ABC DEF0 1234 5678 9ABC DEF0  |.4Vx.....4Vx....| A
DC00:0020  4142 4344 45                             |ABCDE           | A
```

The A/E to the right of the dump indicates whether the characters displayed are the ASCII or EBCDIC translations of the data dumped.

Error Messages:

Invalid Segment Specification.
Invalid Offset Specification.
Invalid Length Specification.
Address specified wraps addressability boundary.

9.18 MW - Display One Word of CPU Board Memory

Use the MW command to display one word of CPU board memory. Only one word may be displayed at a time. Values may be entered in either upper or lower case, and leading zeros are not required.

Format:

MW	seg:offs

Where:

seg Specifies the memory segment value, in hex.

offs Specifies the memory offset, in hex.

Example:

```
mw dc00:0
```

This command displays the word that is in CPU memory at the given location. A response might be:

```
w  DC00:0000  0120
```

The 'w' indicates that the display is of a Word, and not 2 Bytes. The word is not byte-reversed as it is displayed.

Error Messages:

Invalid Segment Specification.
Invalid Offset Specification.

9.19 N - Send Notes to Attached Terminals

Use the N command to display notes on all or one attached terminal.

Format:

N	a msg
N	device msg

Where:

a Is the letter 'A'. This dispatches notes to all attached terminals

device Is the decimal device number of one terminal to receive the message

msg Is a message up to 70 characters long

Notes:

1. After receiving the note the terminal user must key in the MASTER RESET sequence (usually a control-G) to return to his host session.

2. Only the first 70 characters of the message following the command will be sent to the devices. Long messages must be sent via several NOTE commands.

3. When notes are sent to all terminals, output is suppressed on hardcopy devices and devices operating in the IBM 7171 transparency mode. Sending a note to a device that is operating in IBM 7171 transparency mode generates an error. Sending a note to a specific device that is in hardcopy mode is allowed.

Example:

```
n a This is a note to all terminals.
```

This command sends the above message to all terminals attached to the IBM 7171 at the time the command was issued.

Error Messages:

Invalid Device Number.
Device Not Attached.

9.20 O - Outword Command

Use the O command to write a word to the specified I/O port. The port value and word are specified in hex. The values may be entered in either upper or lower case, and leading zeros are not required.

Format:

O	port val

Where:

port Specifies the I/O port to be written to. The port number is specified in hex, and the response is in hex.

val Specifies the value to be written, given in hex.

Example:

```
O ffc0 1234
```

This command writes the hex value 1234 to I/O port FFC0.

Error Messages:

Invalid I/O Port Number Specification.
Invalid Output Data Given.

9.21 P - Display Port Configuration

Use the P command to display the IBM 7171 current port configuration.

Format:

P	
P	tc

Where:

tc Indicates a request to display the port configuration of a specific TC.

Example:

```
p
```

This command requests a display of the IBM 7171 port configuration. A response might be:

```
tc 0   0-036C IBM3101   1-Unattached   2-046B HARDCOPY   3-Unattached
tc 0   4-04AA IBM3101   5-Unattached   6-Unattached      7-Unattached
tc 1   8-Unattached     9-Unattached  10-Unattached     11-Unattached
tc 1  12-059D IBM3101  13-Unattached  14-Unattached     15-Unattached
```

The TC on which the ports exist is listed at the start of the line. For each port there is the following information: dvc,addr,type

dvc Is the device number of that port. This is a decimal value, and is to be used for all other device-specific commands, such as Configure, Force, and Note.

addr Indicates the segment address of this port's data control block.

type Indicates what type of terminal, if known, is attached to this port.

Notes:

1. If the terminal is at the 'ENTER TERMINAL TYPE' message, and is not yet initialized, this field will be left blank.
2. Any port listed as UNATTACHED indicates that either nothing is plugged into that port, or the terminal is powered off.
3. If the terminal type was predefined for a specific port but the terminal type can not be determined from the address specified, the word INVALID will be displayed.
4. Any invalid characters that are typed in response to the ENTER TERMINAL TYPE message will be displayed in this field. This is to aid in locating certain terminals. For instance, if a unique character sequence was typed in response to the ENTER TERMINAL TYPE message, such as "xxxxx," the "xxxxx"

would show up in the Ports response, and the device number of the terminal in question could be identified.

Error Messages:

Invalid TC board specified.
TC not inserted.

9.22 PW - Change Maintenance Passwords

Use the PW command to change the IBM 7171 Maintenance Facility passwords.

Format:

PW	o password
PW	z password

Where:

o Indicates that this request is to change the Maintenance password.

z Indicates that this request is to change the ZAP password.

Example:

```
pw o password
```

This command changes the maintenance password to 'PASSWORD'.

```
pw z password
```

This command changes the zap password to 'PASSWORD'. Refer to "9.5 How to Change the Password and Parse Tables" on page 9-3 in this chapter for more complete description of the passwords.

Error Messages:

Invalid password type given.
Invalid password - Password may not begin with blank.

9.23 S - Store to CPU Board Memory

Use the S command to store data to CPU board memory. Up to one line of data may be stored at a time. The bytes are stored sequentially, one byte at a time, starting at the given address.

Format:

S	seg:offs [bytebyte . . .] [byte]
S	seg:offs [bytebyte . . .]
S	seg:offs string

Where:

seg Specifies the memory segment value, in hex.

offs Specifies the memory offset, in hex.

bytebyte Means the representation of two bytes, i.e. 12EF.

byte Means the representation of one byte, i.e. 2E.

string Means a string of characters surrounded by single quotes. The ASCII characters themselves are stored sequentially, starting at the given address. All characters between the first and last quote will be stored. Any single quotes inside the string will also be stored.

Notes:

1. Only one line of characters may be stored. The final quote must appear on the current command line.

2. A check of all given data is done before any data is actually stored in memory. If an error occurs, NO data will have been stored.

Example:

```
s DC00:200 4142 4344 45
```

Stores the 5 bytes given in CPU memory starting at segment DC00 and offset 200.

```
s DC00:200 45
```

Stores the 1 byte given in CPU memory starting at segment DC00 and offset 200.

```
s DC00:200 45 4142
```

Is illegal and an error message will be displayed.

```
s DC00:200 'Isn't it a beautiful day?'
```

Stores the phrase "Isn't it a beautiful day?" in CPU memory starting at segment DC00 and offset 200. Note that the apostrophe inside the phrase is also stored.

Error Messages:

Invalid Segment Specification.
Invalid Offset Specification.
Invalid Storage Byte Format.
No closing delimiter found on ASCII store.

9.24 SW - Store One Word into CPU Board Memory

Use the SW command to store one word into CPU board memory. Only one word at a time may be stored. Values may be entered in either upper or lower case, and leading zeros are not required.

Format:

SW	seg:offs word

Where:

seg Specifies the memory segment value, in hex.

offs Specifies the memory offset, in hex.

word Is the one word that you wish to store.

Example:

```
sw 8000:40b8 0120
```

This command stores the word specified at the given memory location. No response is given.

Error Messages:

Invalid Segment Specification.
Invalid Offset Specification.
Invalid Storage Word.

9.25 T - Display Terminal Type Information

Use the T command to display parameters relating to a specific terminal type. Terminal name may be entered in either upper or lower case, and space filling is not required.

Format:

T	term

Where:

term Is the terminal type for which information is requested.

Example:

```
t ibm3101
```

This command requests a display of parameters relating to the IBM3101 terminal type. The parameters are shown not byte reversed. A typical response would be:

```
All terminal offset values are relative to segment DC00:
Address of TDT: 52F2
Flags:        0000  Input Parse Tables: 534E  Output Parse Tables: 5304
Cursor Base: 0020  Host Trans Tables:  425E  Term Trans Tables:   4C72
Reset Chars: 5172  Transmit delay:     0000  Attribute Params:    517B
```

Flags Is the actual value of this terminal's flags.

Input Parse Tables Pointer to the first level input parse table.

Output Parse Tables Pointer to the Control String Pointer List.

Cursor Base Cursor addressing base. (i.e. position 1 = 1 + cursor base)

Host Trans Tables Pointer to Host EBCDIC to ASCII table.

Term Trans Tables Pointer to Terminal ASCII to EBCDIC table.

Reset Chars Pointer to IBM 7171 Reset Characters string.

Transmit delay Delay (in milliseconds) in output of initialization string.

Attribute Params Pointer to "Set Graphic Rendition" string.

For a complete description of these values, refer to "Terminal Header Information" on page 4-30 in Chapter 4 of this document.

Error Messages:

Invalid Terminal Type given.

9.26 TM - Display Terminal Controller (TC) Board Memory

Use the TM command to display memory of any TC board. Up to hex FF bytes of TC memory may be displayed at one time. Values may be entered in either upper or lower case, and leading zeros are not required.

Format:

TM	n seg:offs len
TM	n seg:offs

Where:

n Specifies which TC (0-7) boards memory is to be displayed.

seg Specifies the memory segment value, in hex.

offs Specifies the memory offset, in hex.

len Indicates the number of bytes to be displayed, in hex. If length is omitted, one byte will be displayed. If length specified is larger than hex FF, FF will be the default.

Example:

```
tm 2 0:1a0 12
```

This command requests a display of hex 12 bytes of TC 2 memory starting at segment 0 and offset 1A0. A response might be:

```
t2 0000:01A0  1234 5678 9ABC DEF0 1234 5678 9ABC DEF0  |.4Vx.....4Vx....| A
t2 0000:01B0  1234                                     |.4              | A
```

The 't2' indicates that memory from TC 2 is being displayed. The A/E to the right of the dump indicates whether the characters displayed are the ASCII or EBCDIC translations of the data dumped.

Error Messages:

Invalid segment specification.
Invalid offset specification.
Invalid length specification.
Invalid TC board specification
TC not inserted.
Address specified wraps addressability boundary.

9.27 TS - Store to Terminal Controller (TC) Board Memory

Use the TS command to store to TC memory. Only one byte at a time may be stored to TC memory. Values may be entered in either upper or lower case, and leading zeros are not required.

Format:

TS	n seg:offs byte

Where:

n Specifies which TC board is to be stored to.

seg Specifies the memory segment value, in hex.

offs Specifies the memory offset, in hex.

byte Is the one byte that will be stored.

Example:

```
ts 0 0:1a0 b3
```

This command stores the value B3 in TC 0 memory at segment 0 and offset 1A0. No response is given.

Error Messages:

Invalid segment specification.
Invalid offset specification.
Invalid storage byte format.
Invalid TC board specified.
TC not inserted.

9.28 X - Toggle ASCII/EBCDIC Output Display

Use the X command to toggle character display on memory dumps between ASCII and EBCDIC. The output type is determined by examining the trailing character on the output dump. An 'A' indicates ASCII output is selected, an 'E' indicates EBCDIC output is selected

Format:

x	

Example:

```
x
```

This command toggles output display format. Responses are:

```
ASCII output selected
EBCDIC output selected
```

9.29 = - Invoke Calculator Functions

Use the = command to invoke calculator functions. The '=' command is actually three commands: '=S', '=T', and '=X'. The character after the '=' indicates what default output to select for the answer, ASCII, Decimal, or Hex, respectively.

Format:

=S	val [. . .op val]
=T	val [. . .op val]
=X	val [. . .op val]

Where:

val Specifies any value, or operand. An operand can be specified in two ways:

num Where 'num' is interpreted to be of the output default type selected in the command, either one ASCII character, one decimal value where legal characters are 0-9, or hexadecimal where legal characters are 0-9 and A-F.

Nnum Where N is the character 'S', 'T', or 'X'. This character prefix to the value overrides the default output type given on the command line. The next value is assumed to have a type N, where N is as above. Num is assumed to be either an ASCII character, a decimal value, or a hexadecimal value, respectively.

op Specifies the operation to be performed. Only the four basic operations are supported. They are:

+ Adds the value to the left of the plus to the value on the right of the plus.

- Subtracts the value immediately to the right of the minus from the value on the left of the minus.

* Multiplies the values on the right and left of the '*' together.

/ Divides the value on the left of the '/' by the value on the right of the '/'.

Warning: Order of operations is LEFT to RIGHT. NO operation precedence exists. i.e. the problem 2+3*10 is calculated as 50, NOT 32.

Notes:

1. NO spaces are allowed in the arithmetic string. Any space is considered a termination of the string.

2. Precision is limited to 16 bits. It is the user's responsibility to account for overflows and carries. All ASCII input is limited to ONE character per operand, and ASCII output is limited to two character output.

3. The high order bit is masked off for ASCII output. It is the user's responsibility to recognize when this has occurred. Also, all characters from X'00' to X'1F' are represented by a ':' when they are output.

Example:

```
=x t64
0040
Ok
```

This command interprets 64 to be in decimal, and displays the answer in HEX. It has the effect of translating decimal to hex. Similarly, the command **"=x s#"** will translate the ASCII character '#' into a hexadecimal value.

```
=t s0+xa-8/10
5
Ok
```

This string took the ASCII character 0 (decimal 48) added to it hex A (decimal 10) subtracted 8, and then divided by 10. Since no type specifier preceded the 8 or the 10, both were taken to be decimal numbers since the default output type is decimal.

Error Messages:

Improper Arithmetic String.

9.30 System Error Messages

Following is a list of System Error Messages, and their associated parameters:

Received unsolicited DMA.

- **P0** Command from TC, after one shift-left.
- **P1** SCCOUT, STCCOUT processing flags.
- **P2** TC DMA pending, SCRN buffer status flag.

Received data or DMA for unconnected line.

- **P0** Command from TC, after one shift-left.
- **P1** SCCOUT, STCCOUT processing flags.
- **P2** TC DMA pending, SCRN buffer status flag.

CPU sent TC an unsupported command.

- **P0** Command from TC, after one shift-left.
- **P1** SCCOUT, STCCOUT processing flags.
- **P2** TC DMA pending, SCRN buffer status flag.

KYBD parsed key sequence that generated an invalid routine number.

- **P0** Register SI, Address of SEQ NODE.
- **P1** Register AX before shifting. SEQNCALL that triggered error. AL= routine #.
- **P2** AC2WORK, Character that triggered invalid call, last character in input sequence.

Out of RAM space.

- **P0** Amount of memory requested.
- **P1** SCCOUT, STCCOUT processing flags.
- **P2** TC DMA pending, SCRN buffer status flag.

TC DMA timed out.

- **P0** Data Segment (DS) register for Device.
- **P1** Start address of output buffer.

P2 Count.

CC was interrupted while previous DMA was pending.

P0 Command from CC.

P1 Device #, SCRN buffer status flag.

P2 Command from CC of DMA that is pending.

CC and CPU "chained" miscommunication occurred.

P0 Count of data discarded.

P1 Device #, SCRN buffer status flag.

P2 Command from CC of DMA that is pending.

CPU had more than 2 status requests for a device.

P0 Status, Sense which could not be sent.

P1 Status, Sense of first in queue.

P2 Status, Sense of second in queue.

CC 3270 data stream buffer too small. Change default.

P0 Count of data discarded.

P1 Status, Sense of first in queue.

P2 Status, Sense of second in queue.

Multiple DMA requests were made to the CC.

P0 Length for requested DMA.

P1 Previous DMA command.

P2 Device #, SCRN buffer status flag.

CC DMA timed out.

P0 Length for requested DMA.

P1 Previous DMA command.

P2 Device #, SCRN buffer status flag.

Invalid or unexpected interrupt.

P0 Flags Register.

P1 Code Segment Register.

P2 Program Counter.

Note: The device number listed is the invalid interrupt number that caused the error.

Read or Read Modified found DMA buffer marked full.

P0 Port command from CC, SCRN flag.

P1 SCRN buffer status flag, Device number.

P2 AC2 for write datastream.

Write or Erase/Write found DMA buffer not marked full.

P0 DMA command from CC, SCRN flag.

P1 SCRN buffer status flag, Device number.

P2 AC2 for write datastream.

SCRN received illegal command from CC.

P0 DMA command from CC, port command from CC.

P1 SCRN flag, SCRN buffer status.

P2 AC2 for write datastream.

9.31 General Notes

1. The area used for gathering CC and TC data is initialized to X'DD'. Any bytes that appear as X'DD' may have been the result of miscommunication.

2. Display of TC and CC memory, and sending of notes all utilize the same work area. Care must be taken to make sure that one function is finished before initiating another.

Appendix A. ASCII and EBCDIC Data Conversion Tables

This appendix gives the tables used for translating the data transferred between the IBM 7171 and the host computer and between the IBM 7171 and ASCII terminals.

A.1 How to Read These Tables

To translate a byte of data using these tables, find the most significant digit of the byte in the left hand column of indices, then the least significant digit in the top row of indices. The row and column that each respective index denotes intersect, giving the desired value. An example of a table lookup follows.

		Least Significant Digit									
		0	....	4	5	6	7	8	9	....	F
	0	01	....	DF	02	DF	DF	0C	DF	....	DF
	.	.		.					.		.
	.	.		.					.		.
	.	.		.					.		.
Most	6	2D	....	DF	DF	DF	DF	DF	DF	....	3F
Significant	7	DF	....	DF	DF	DF	DF	DF	60	....	22
Digit	8	DF	....	64	65	66	67	68	69	....	DF
	9	DF	....	6D	6E	6F	70	71	72	....	DF
	A	DF	....	75	76	77	78	79	7A	....	DF
	.	.		.					.		.
	.	.		.					.		.
	.	.		.					.		.
	F	30	....	34	35	36	37	38	39	....	DF

Figure A-1. Sample Translation Table

For example, to translate a X'96' using this sample table, first find the '9' in the left hand column, then find the '6' in the top row. The byte at the place where the '9' row and the '6' column intersect is X'6F', therefore this is the translated value of X'96'.

A.2 Host Tables for 3277 Device Types

These tables are used if the current host SYSGEN supports a 3277 terminal on the given line, and the appropriate 3277 override has been made in Non-Volatile RAM (NV-RAM). Refer to "Chapter 4. Customizing IBM 7171 Tables" for customizing tables in NV-RAM.

A.2.1 3277 Basic Translate Table for Write Data Streams

This table translates host EBCDIC data to an internal ASCII representation. The mapping ranges are:

```
X'00'-X'40': 3270 Data Stream Orders
X'41'-X'FF': EBCDIC Characters
```

Illegal EBCDIC characters are denoted in the table by X'DF'.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	01	DF	DF	00	DF	02	DF	DF	DF	DF	DF	DF	83	84	DF	DF
1	DF	03	04	05	DF	81	DF	DF	DF	82	DF	DF	09	06	0A	DF
2	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	0B	DF	DF	DF	DF
3	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	07	DF	DF	DF
4	08	DF	DF	DF	DF	DF	DF	DF	DF	DF	5C	2E	3C	28	2B	7C
5	26	DF	DF	DF	DF	DF	DF	DF	DF	DF	21	24	2A	29	3B	5E
6	2D	2F	DF	DF	DF	DF	DF	DF	DF	DF	86	2C	25	5F	3E	3F
7	DF	5E	DF	DF	DF	DF	DF	DF	DF	60	3A	23	40	27	3D	22
8	DF	61	62	63	64	65	66	67	68	69	B9	DF	DE	B3	A4	BD
9	AC	6A	6B	6C	6D	6E	6F	70	71	72	B8	BA	DF	AF	DF	BB
A	A0	7E	73	74	75	76	77	78	79	7A	A3	B6	A2	5B	BE	AA
B	A1	A5	A9	B2	B7	DF	DB	DD	BF	DF	A7	A8	AE	5D	B1	AD
C	7B	41	42	43	44	45	46	47	48	49	DF	DF	DF	DF	DF	DF
D	7D	4A	4B	4C	4D	4E	4F	50	51	52	DF	DF	DF	DF	DF	DF
E	85	DF	53	54	55	56	57	58	59	5A	DF	DF	DF	DF	DF	DF
F	30	31	32	33	34	35	36	37	38	39	DF	DF	DF	DF	DF	DF

A.2.2 3277 Attribute and APL Byte Table for Write Data Streams

This table translates host attribute or Data Analysis (APL) characters (bytes that follow the Start Field X'1D' order) to an internal representation.

Illegal attribute or APL characters are denoted in the table by X'DF'.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	00
1	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
2	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
3	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
4	E0	E1	DF	DF	E2	E3	DF	DF	E4	E5	95	94	E6	E7	90	91
5	E8	E9	DF	DF	EA	EB	DF	DF	EC	ED	98	9B	EE	EF	9D	9C
6	F0	F1	DF	DF	F2	F3	DF	DF	F4	F5	B0	AB	F6	F7	DF	DF
7	F8	F9	DF	DF	FA	FB	DF	DF	FC	FD	DF	B4	FE	FF	DF	DF
8	DF	C1	C2	C3	C4	C5	C6	C7	C8	C9	DF	DF	DF	DF	DF	DF
9	87	CA	CB	CC	CD	CE	CF	D0	D1	D2	DF	C0	DF	DF	DF	DF
A	88	DF	D3	D4	D5	D6	D7	D8	D9	DA	DF	DF	DF	BC	DC	DF
B	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
C	E0	E1	B5	A6	E2	E3	8D	93	E4	E5	DF	DF	E6	E7	DF	DF
D	E8	E9	9F	96	EA	EB	8E	8F	EC	ED	DF	DF	EE	EF	DF	DF
E	F0	F1	92	97	F2	F3	9A	99	F4	F5	DF	DF	F6	F7	DF	DF
F	F8	F9	DF	9E	FA	FB	DF	DF	FC	FD	DF	DF	FE	FF	DF	DF

A.2.3 3277 Basic Translate Table for Read Data Streams

This table translates the internal ASCII representation of a character to EBCDIC. The mapping ranges are:

```
X'00'        : Null Character
X'01'-X'1F': Miscellaneous Control Codes
X'20'-X'7E': Normal ASCII Graphic Characters
X'7F'-X'8C': Special Characters (FM, DUP, NL, EM, FF, CR)
X'8D'-X'DE': 3278 APL Characters
X'DF'        : Returns X'60' for 3278 Illegal Character
X'E0'-X'FF': Attribute Characters
```

A X'00' in this table (except for null) means that the character is either an attribute byte or an APL compound character, and should be translated on the appropriate table.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	01	02	03	37	2D	2E	2F	16	05	15	0B	0C	26	0E	0F
1	10	11	12	13	3C	3D	32	26	18	19	3F	27	1C	1D	1E	1F
2	40	5A	7F	7B	5B	6C	50	7D	4D	5D	5C	4E	6B	60	4B	61
3	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	7A	5E	4C	7E	6E	6F
4	7C	C1	C2	C3	C4	C5	C6	C7	C8	C9	D1	D2	D3	D4	D5	D6
5	D7	D8	D9	E2	E3	E4	E5	E6	E7	E8	E9	AD	4A	BD	5F	6D
6	79	81	82	83	84	85	86	87	88	89	91	92	93	94	95	96
7	97	98	99	A2	A3	A4	A5	A6	A7	A8	A9	C0	4F	D0	A1	1E
8	1C	15	19	0C	0D	E0	6A	00	00	60	60	60	60	00	00	00
9	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
A	A0	B0	AC	AA	8E	B1	00	BA	BB	B2	AF	00	90	BF	BC	9D
B	00	BE	B3	8D	00	00	AB	B4	9A	8A	9B	9F	00	8F	AE	B8
C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
D	00	00	00	00	00	00	00	00	00	00	00	B6	00	B7	8C	60
E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

A.2.4 3277 Attribute and APL Table for Read Data Streams

This table translates the internal representations of an attribute or 3277 Data Analysis (APL) byte to EBCDIC, which will then follow a Start Field X'1D' order. The mapping ranges are:

```
X'00'-X'7E': Unused
X'7F'-X'8C': Special Characters
X'8D'-X'DF': 3277 APL Characters
X'E0'-X'FF': Attribute Characters
```

Illegal attribute or APL bytes are denoted in the table by X'FF'.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
1	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
2	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
3	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
4	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
5	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
6	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
7	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
8	FF	FF	FF	FF	FF	FF	FF	90	A0	FF	FF	FF	FF	C6	D6	D7
9	4E	4F	E2	C7	4B	4A	D3	E3	5A	E7	E6	5B	5F	5E	F3	D2
A	FF	FF	FF	FF	FF	FF	C3	FF	FF	FF	FF	6B	FF	FF	FF	FF
B	6A	FF	FF	FF	7B	C2	FF	FF	FF	FF	FF	FF	AD	FF	FF	FF
C	9B	81	82	83	84	85	86	87	88	89	91	92	93	94	95	96
D	97	98	99	A2	A3	A4	A5	A6	A7	A8	A9	FF	AE	FF	FF	FF
E	40	C1	C4	C5	C8	C9	4C	4D	50	D1	D4	D5	D8	D9	5C	5D
F	60	61	E4	E5	E8	E9	6C	6D	F0	F1	F4	F5	F8	F9	7C	7D

A.3 Host Tables for 3278 Device Types

These are the default tables used by the IBM 7171, that is, no explicit override to use 3277 tables has been made in Non-Volatile RAM.

A.3.1 3278 Basic Translate Table for Write Data Streams

This table translates host EBCDIC data to an internal ASCII representation. The mapping ranges are:

```
X'00'-X'40': 3270 Data Stream Orders
X'41'-X'FF': EBCDIC Characters
```

Illegal EBCDIC characters are denoted in the table by X'DF'.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	01	DF	DF	00	DF	02	DF	DF	0C	DF	DF	DF	83	84	DF	DF
1	DF	03	04	05	DF	81	DF	DF	DF	82	DF	DF	09	06	0A	DF
2	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	0B	DF	DF	DF	DF
3	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	07	DF	DF	DF
4	08	DF	DF	DF	DF	DF	DF	DF	DF	DF	5C	2E	3C	28	2B	7C
5	26	DF	DF	DF	DF	DF	DF	DF	DF	DF	21	24	2A	29	3B	5E
6	2D	2F	DF	DF	DF	DF	DF	DF	DF	DF	86	2C	25	5F	3E	3F
7	DF	DF	DF	DF	DF	DF	DF	DF	DF	60	3A	23	40	27	3D	22
8	DF	61	62	63	64	65	66	67	68	69	DF	DF	DF	DF	DF	DF
9	DF	6A	6B	6C	6D	6E	6F	70	71	72	DF	DF	DF	DF	DF	DF
A	DF	7E	73	74	75	76	77	78	79	7A	DF	DF	DF	5B	DF	DF
B	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	5D	DF	DF
C	7B	41	42	43	44	45	46	47	48	49	DF	DF	DF	DF	DF	DF
D	7D	4A	4B	4C	4D	4E	4F	50	51	52	DF	DF	DF	DF	DF	DF
E	85	DF	53	54	55	56	57	58	59	5A	DF	DF	DF	DF	DF	DF
F	30	31	32	33	34	35	36	37	38	39	DF	DF	DF	DF	DF	DF

A.3.2 3278 Attribute Byte Table for Write Data Streams

This table translates host attribute bytes (bytes that follow the Start Field X'1D' order) to an internal representation.

Illegal attribute characters are denoted in the table by X'DF'.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
1	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
2	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
3	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
4	E0	E1	DF	DF	E2	E3	DF	DF	E4	E5	DF	DF	E6	E7	DF	DF
5	E8	E9	DF	DF	EA	EB	DF	DF	EC	ED	DF	DF	EE	EF	DF	DF
6	F0	F1	DF	DF	F2	F3	DF	DF	F4	F5	DF	DF	F6	F7	DF	DF
7	F8	F9	DF	DF	FA	FB	DF	DF	FC	FD	DF	DF	FE	FF	DF	DF
8	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
9	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
A	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
B	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
C	E0	E1	DF	DF	E2	E3	DF	DF	E4	E5	DF	DF	E6	E7	DF	DF
D	E8	E9	DF	DF	EA	EB	DF	DF	EC	ED	DF	DF	EE	EF	DF	DF
E	F0	F1	DF	DF	F2	F3	DF	DF	F4	F5	DF	DF	F6	F7	DF	DF
F	F8	F9	DF	DF	FA	FB	DF	DF	FC	FD	DF	DF	FE	FF	DF	DF

A.3.3 3278 APL Table for Write Data Streams

This table translates host APL characters (bytes that follow the Graphic Escape X'08' order) to their internal representations.

Illegal APL characters are denoted in the table by X'DF'.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
1	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
2	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
3	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF
4	DF	C1	C2	C3	C4	C5	C6	C7	C8	C9	DF	DF	DF	DF	DF	DF
5	DF	CA	CB	CC	CD	CE	CF	D0	D1	D2	DF	DF	DF	DF	DF	DF
6	DF	DF	D3	D4	D5	D6	D7	D8	D9	DA	DF	DF	DF	DF	DF	DF
7	DF	B0	A6	DF	DF	DF	DF	DF	AB	DF	DF	DF	DF	DF	DF	DF
8	B4	DF	DF	DF	DF	DF	DF	DF	DF	DF	B9	B5	DE	B3	A4	BD
9	AC	DF	DF	DF	DF	DF	DF	DF	DF	DF	B8	BA	C0	AF	DF	BB
A	A0	DF	DF	DF	DF	DF	DF	DF	DF	DF	A3	B6	A2	89	BE	AA
B	A1	A5	A9	B2	B7	DF	DB	DD	BF	DF	A7	A8	AE	8A	B1	AD
C	87	DF	DF	DF	DF	DF	BC	DF	DF	DF	95	94	DF	90	DF	91
D	88	DF	DF	DF	DF	DF	DC	DF	DF	DF	9F	96	8E	8F	98	9B
E	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	9D	9C	DF	92	97	9A
F	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	DF	8D	9E	93	99	DF

A.3.4 3278 Basic Translate Table for Read Data Streams

This table translates the internal ASCII representation of a character to EBCDIC. The mapping ranges are:

```
X'00'        : Null Character
X'01'-X'1F': Miscellaneous Control Codes
X'20'-X'7E': Normal ASCII Graphic Characters
X'7F'-X'8C': Special Characters (FM, DUP, NL, EM, FF, CR)
X'8D'-X'DE': 3278 APL Characters
X'DF'        : Returns X'60' for 3278 Illegal Character
X'E0'-X'FF': Attribute Characters
```

A X'00' in this table (except for null) means that the character is either an attribute byte or an APL compound character, and should be translated on the appropriate table.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	01	02	03	37	2D	2E	2F	16	05	15	0B	0C	0D	0E	0F
1	10	11	12	13	3C	3D	32	26	18	19	3F	27	1C	1D	1E	1F
2	40	5A	7F	7B	5B	6C	50	7D	4D	5D	5C	4E	6B	60	4B	61
3	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	7A	5E	4C	7E	6E	6F
4	7C	C1	C2	C3	C4	C5	C6	C7	C8	C9	D1	D2	D3	D4	D5	D6
5	D7	D8	D9	E2	E3	E4	E5	E6	E7	E8	E9	AD	4A	BD	5F	6D
6	79	81	82	83	84	85	86	87	88	89	91	92	93	94	95	96
7	97	98	99	A2	A3	A4	A5	A6	A7	A8	A9	C0	4F	D0	A1	1E
8	1C	15	19	0C	0D	E0	6A	00	00	00	00	60	60	00	00	00
9	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	60
E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

A.3.5 3278 Attribute Byte Table for Read Data Streams

This table translates the internal representations of an attribute byte to EBCDIC, which will then follow a Start Field X'1D' order.

Illegal attribute bytes are denoted in the table by X'FF'.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
1	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
2	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
3	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
4	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
5	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
6	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
7	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
8	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
9	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
A	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
B	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
C	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
D	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
E	40	C1	C4	C5	C8	C9	4C	4D	50	D1	D4	D5	D8	D9	5C	5D
F	60	61	E4	E5	E8	E9	6C	6D	F0	F1	F4	F5	F8	F9	7C	7D

A.3.6 3278 APL Table for Read Data Streams

This table translates the internal representation of an APL character to EBCDIC , which will then follow a Graphic Escape X‘08’ order.

Illegal APL characters are denoted in the table by X‘60’.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
1	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
2	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
3	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
4	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
5	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
6	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
7	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
8	60	60	60	60	60	60	60	C0	D0	AD	BD	60	60	FB	DC	DD
9	CD	CF	ED	FD	CB	CA	DB	EE	DE	FE	EF	DF	EB	EA	FC	DA
A	A0	B0	AC	AA	8E	B1	72	BA	BB	B2	AF	78	90	BF	BC	9D
B	71	BE	B3	8D	80	8B	AB	B4	9A	8A	9B	9F	C6	8F	AE	B8
C	9C	41	42	43	44	45	46	47	48	49	51	52	53	54	55	56
D	57	58	59	62	63	64	65	66	67	68	69	B6	D6	B7	8C	60
E	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60
F	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60	60

A.4 Terminal Translate Tables

These tables are used to translate data between ASCII based terminals and the IBM 7171 extended ASCII code used in the representation of the terminal screen image. Refer to "Chapter 4. Customizing IBM 7171 Tables" for customizing tables in NV-RAM.

Special handling routines are used to convert the data when the byte in the table has its high order bit on. The special handling codes are indices into a branch table and are defined as follows:

```
X'FF'     Character is only in other (APL/Non-APL) set
X'FE'     Character is underscored alphabetic
X'FD'     Character is general overstrike
X'FC'     Character is illegal graphic
X'FB'     Character is attribute byte
X'FA'     Character is lowercase (APL tables only)
X'F9'     Character is printer graphic
```

A.4.1 Normal (Non-APL) ASCII Output Translate Table

This table translates a character from the IBM 7171 screen image buffer represented by extended ASCII code to normal ASCII code for output to the terminal.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	20	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC
1	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC
2	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
3	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
4	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
5	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
6	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
7	70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	3B
8	2A	F9	F9	F9	F9	5C	7C	7B	7D	FF	FF	FC	FC	FF	FF	FF
9	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
A	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
B	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
C	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
D	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FC
E	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB
F	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB

A.4.2 Normal ASCII Input Translate Table

This table translates normal terminal input data. A X'00' byte indicates a control string introducer.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F
3	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
4	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
5	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
6	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
7	70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	00

A.4.3 Typewriter Paired APL Output Translate Table

This table translates the internal ASCII representation of a character to typewriter paired ASCII code for output to a terminal.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	20	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC
1	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC
2	20	FF	FF	FF	7E	FF	FF	4B	3A	22	50	2D	2C	5F	2E	2F
3	30	31	32	33	34	35	36	37	38	39	3E	3C	23	25	26	51
4	FF	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
5	70	71	72	73	74	75	76	77	78	79	7A	FF	3F	FF	FF	46
6	FF	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA
7	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	7B	4D	7D	FF	3C
8	50	F9	F9	F9	F9	3F	4D	7B	7D	3B	27	FC	FC	FD	FD	FD
9	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD
A	40	41	42	43	44	45	21	47	48	49	4A	28	4C	4D	4E	4F
B	29	2A	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	2B
C	60	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE
D	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	3D	7C	3F	24	FC
E	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB
F	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB

A.4.4 Typewriter Paired APL Input Translate Table

This table translates typewriter paired APL terminal input data.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2	20	A6	29	3C	DE	3D	3E	8A	AB	B0	B1	BF	2C	2B	2E	2F
3	30	31	32	33	34	35	36	37	38	39	28	89	3B	DB	3A	DD
4	A0	A1	A2	A3	A4	A5	5F	A7	A8	A9	AA	27	AC	AD	AE	AF
5	2A	3F	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	2D
6	C0	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
7	50	51	52	53	54	55	56	57	58	59	5A	7B	DC	7D	24	00

A.4.5 Bit Paired APL Output Translate Table

This table translates the internal ASCII representation of a character to bit paired ASCII code for output to a terminal.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	20	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC
1	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC	FC
2	20	FF	FF	FF	7C	FF	FF	4B	2B	2A	50	2D	2C	3D	2E	2F
3	30	31	32	33	34	35	36	37	38	39	3E	3C	23	25	27	51
4	FF	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F
5	70	71	72	73	74	75	76	77	78	79	7A	FF	3F	FF	FF	46
6	FF	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA
7	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	FA	5D	4D	7D	FF	3C
8	50	F9	F9	F9	F9	3F	4D	5D	7D	3B	3A	FC	FC	FD	FD	FD
9	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD	FD
A	22	41	42	43	44	45	21	47	48	49	4A	29	4C	4D	4E	4F
B	5F	28	52	53	54	55	56	57	58	59	5A	40	5B	60	26	7E
C	5C	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE
D	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	FE	5E	7B	3F	24	FC
E	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB
F	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB	FB

A.4.6 Bit Paired APL Input Translate Table

This table translates bit paired APL terminal input data.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
1	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
2	20	A6	A0	3C	DE	3D	BE	3E	B1	AB	29	28	2C	2B	2E	2F
3	30	31	32	33	34	35	36	37	38	39	8A	89	3B	2D	3A	DD
4	BB	A1	A2	A3	A4	A5	5F	A7	A8	A9	AA	27	AC	AD	AE	AF
5	2A	3F	B2	B3	B4	B5	B6	B7	B8	B9	BA	BC	C0	7B	DB	B0
6	BD	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
7	50	51	52	53	54	55	56	57	58	59	5A	DC	24	7D	BF	00

Appendix B. IBM 7171 Supplied Terminal Definition Tables

B.1 Functions of Specific ASCII Terminals

This section shows what ASCII character sequences invoke IBM 7171 and host functions. Most of this information is given in tables, which contain the key sequences for the following ASCII display terminals:

IBM 3101	see "B.6 IBM 3101 Terminal" on page B-4
IBM Personal Computer Running in 3101 Emulation Mode	see "B.7 IBM Personal Computer Running in 3101 Emulation Mode" on page B-9
Televideo TVI 912	see "B.8 TVI-912 and TVI-920 Terminal" on page B-16
Televideo TVI 920	see "B.8 TVI-912 and TVI-920 Terminal" on page B-16
Televideo TVI 950	see "B.9 TVI-950 Terminal" on page B-21
Lear Siegler ADM 31	see "B.10 ADM-31 Terminal" on page B-27
Lear Siegler ADM 3A	see "B.11 ADM-3A Terminal" on page B-32
Digital Equipment VT 100	see "B.12 VT-100 Terminal" on page B-37
Datamedia 1520	see "B.13 DM-1520 and DM-1521 Terminal" on page B-43
Datamedia 1521	see "B.13 DM-1520 and DM-1521 Terminal" on page B-43
Datamedia Elite 3045	see "B.14 DM-3045 Terminal" on page B-48

In addition, there are tables which describe basic functions of a typical ASCII typewriter terminal without specifying any brand or model. For:

TYPETERM	see "B.15 TYPETERM Typewriter Terminal" on page B-53

B.2 Table Description

The key settings are shown in three tables for each type of terminal:

Table 1 - Control and Cursor Movement Keys show the control keys for 3270 emulated and extended functions.

Table 2 - Program Functions Keys show regular 3270 PF key functions.

Table 3 - Setup Functions show setup functions for the extended functions.

The tables for each terminal type consist of three columns, which are arranged as shown in Figure B-1 on page B-1.

Column 1	**Column 2**	**Column 3**
3270 Emulated or Extended Function	Corresponding ASCII Code Sequences	Corresponding Terminal Key Sequence Required
Shows the 3270 or extended function that can be initiated from the keyboard of the ASCII terminal.	Contains the ASCII code sequences that initiate the functions in column 1.	Lists the keys or key sequences that correspond to the ASCII codes in column 2. These invoke the function in column 1.

Figure B-1. Table Layout for ASCII Terminal Keyboard Functions

Notes:

1. *Columns 1 and 2 reflect the contents of the IBM 7171 supplied Terminal Definition Tables that describe the relationship between 3270 or extended functions and the ASCII characters that invoke them.*

2. *Columns 2 and 3 reflect the relationship between the keys on the specific keyboard and the ASCII code sequences generated by each keystroke.*

3. *The information in column 3, which refers to the keyboard layout of the particular terminal, is supplied without any guarantee, since terminals with the same type identification may nevertheless be different. The keys referenced in column 3 produce the related functions for a terminal of the particular terminal type. This column may require changes for a different sub-model of this terminal type. The manufacturer-supplied terminal description will describe which key or key sequence to press to generate the ASCII code specified in column 2.*

B.3 Local Reset and Control Functions

Appended to the three tables describing each terminal type is a short section entitled **Local Reset and Control Functions**. The key sequences described here are routed directly to the IBM 7171 and are not passed to the host. They provide control over the transmission and clearing of data to and from the terminal.

B.4 Installation Hints

At the end of each section installation hints are given for the specific terminal type when operating the IBM 7171. Switches located on the outer cover of the terminal, which may be changed without any tools, are described. For any further customizing, refer to the documentation supplied by the manufacturer of the ASCII device.

B.5 Notational Conventions

In the following tables, an ASCII code will be represented by its displayable character representation if the code corresponds to an uppercase or lowercase letter, a number, or a convenient special character such as . , : / and so on.

If it corresponds to an ASCII control character or a special ASCII character that could produce confusion, a mnemonic notation of the ASCII code is used. These mnemonics are explained in Figure B-2.

00	NULL	18	CAN		
01	SOH	19	EM		
02	STX	1A	SUB	3A	: COLON
03	ETX	1B	ESC ESCAPE	3B	; SEMI
04	EOT	1C	FS	3C	< LESS
05	ENQ WRU	1D	GS	3D	= EQUAL
06	ACK	1E	RS	3E	> GREATER
07	BEL	1F	US	3F	? QUESTION
08	BS	20	SPACE BLANK	40	@ AT
09	HT TAB	21	! EXCLAIM		
0A	LF LINEFEED	22	" DQUOTE		
0B	VT	23	# POUND	5B	[LBRACK
0C	FF FORMFEED	24	$ DOLLAR	5C	\ BSLASH
0D	CR	25	% PERCENT	5D	] RBRACK
0E	SO	26	& AND	5E	^ UPARROW
0F	SI	27	' QUOTE	5F	_ UNDER
10	DLE	28	(LPAREN	60	` ACCENT
11	DC1 XON	29	) RPAREN		
12	DC2	2A	* ASTERISK STAR		
13	DC3 XOFF	2B	+ PLUS	7B	{ LBRACE
14	DC4	2C	, COMMA	7C	¦ BAR
15	NAK	2D	- HYPHEN MINUS	7D	} RBRACE
16	SYN	2E	. PERIOD DOT	7E	~ TILDE
17	ETB	2F	/ SLASH	7F	DEL

Figure B-2. ASCII Character Name Table. This table describes the mnemonics used for ASCII control characters together with certain special ASCII characters and the ASCII code to which they correspond in hexadecimal notation

If a sequence of ASCII characters is required to invoke a particular function, the ASCII characters are separated by commas (","). If different ASCII code characters or character sequences will cause the same function to be executed, these characters or character sequences are separated by the expression "or."

If multiple keys have to be pressed on the keyboard to introduce a particular function, they are again separated by commas (","). If two or even three keys have to be pressed simultaneously, they are concatenated with the word symbol "and."

If different keys, key sequences or key combinations will invoke the same function, they are again separated by using the expression "or."

In the column describing the keys that have to be pressed, normally the exact inscription as it can be found on the keyboard will be used.

B.6 IBM 3101 Terminal

B.6.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Seq.	Corresponding IBM 3101 Key Sequence Required
Enter	BS	←
Clear	ESC, L	CLEAR
Test Request	ESC, J, ESC, W	ERASE EOS, PRINT
Local Print	ESC, W	PRINT
DUP Character	VT	ALT and k
Fieldmark Character	FF	ALT and l
Cursor Select	ESC, K	ERASE INPUT
Redisplay	SYN	ALT and v
Erase Input	ESC, J, ESC, K	ERASE EOS, ERASE INPUT
Erase EOF	ESC, I	ERASE EOL/EOF
Delete Character	DEL	DEL
Toggle Insert Mode	ESC, J, DEL	ERASE EOS, DEL
Field Tab	HT	→|
Field Backtab	ESC, J, HT	ERASE EOS, →|
Column Tab	ESC, J, ESC, C or ESC, HT	ERASE EOS, → ESC, →|
Column Backtab	ESC, J, ESC, D or ESC, CR	ERASE EOS, ← ESC, ↵
Indent	ESC, J, ESC, A	ERASE EOS, ↑
Undent	ESC, J, ESC, B	ERASE EOS, ↓
PA1	ESC, J, COMMA or ESC, J, LESS or ESC, J, z	ERASE EOS, , or ERASE EOS, < or ERASE EOS, z (Note 1)
PA2	ESC, J, PERIOD or ESC, J, GREATER or ESC, J, x	ERASE EOS, . or ERASE EOS, > or ERASE EOS, x (Note 1)
PA3	ESC, J, SLASH or ESC, J, QUESTION or ESC, J, c	ERASE EOS, / or ERASE EOS, ? or ERASE EOS, c (Note 1)
Newline	CR	↵
Home	ESC, H	ALT and |←
Cursor Up	ESC, A	↑
Cursor Down	ESC, B	↓
Cursor Right	ESC, C	→
Cursor Left	ESC, D	←

(Note 1) This character may be typed in as uppercase or lowercase character.

Figure B-3. Control and Cursor Movement Keys for the IBM 3101

B.6.2 Program Function Keys

Function	ASCII-CODE (-Sequence)	IBM 3101 Key
PFK 1	ESC, J, 1	ERASE EOS, 1
PFK 2	ESC, J, 2	ERASE EOS, 2
PFK 3	ESC, J, 3	ERASE EOS, 3
PFK 4	ESC, J, 4	ERASE EOS, 4
PFK 5	ESC, J, 5	ERASE EOS, 5
PFK 6	ESC, J, 6	ERASE EOS, 6
PFK 7	ESC, J, 7	ERASE EOS, 7
PFK 8	ESC, J, 8	ERASE EOS, 8
PFK 9	ESC, J, 9	ERASE EOS, 9
PFK 10	ESC, J, 0	ERASE EOS, 0
PFK 11	ESC, J, HYPHEN	ERASE EOS, -
PFK 12	ESC, J, EQUAL	ERASE EOS, =
PFK 13	ESC, J, EXCLAIM or	ERASE EOS, ! or
	ESC, J, q	ERASE EOS, q (Note 1)
PFK 14	ESC, J, AT or	ERASE EOS, @ or
	ESC, J, w	ERASE EOS, w (Note 1)
PFK 15	ESC, J, POUND or	ERASE EOS, # or
	ESC, J, e	ERASE EOS, e (Note 1)
PFK 16	ESC, J, DOLLAR or	ERASE EOS, $ or
	ESC, J, r	ERASE EOS, r (Note 1)
PFK 17	ESC, J, PERCENT or	ERASE EOS, % or
	ESC, J, t	ERASE EOS, t (Note 1)
PFK 18	ESC, J, UPARROW or	ERASE EOS, ^ or
	ESC, J, y	ERASE EOS, y (Note 1)
PFK 19	ESC, J, AND or	ERASE EOS, & or
	ESC, J, u	ERASE EOS, u (Note 1)
PFK 20	ESC, J, STAR or	ERASE EOS, * or
	ESC, J, i	ERASE EOS, i (Note 1)
PFK 21	ESC, J, LPAREN or	ERASE EOS, (or
	ESC, J, o	ERASE EOS, o (Note 1)
PFK 22	ESC, J, RPAREN or	ERASE EOS,) or
	ESC, J, p	ERASE EOS, p (Note 1)
PFK 23	ESC, J, LBRACK or	ERASE EOS, [or
	ESC, J, RBRACK or	ERASE EOS,] or
	ESC, J, UNDER	ERASE EOS, _
PFK 24	ESC, J, BSLASH or	ERASE EOS, \ or
	ESC, J, BAR or	ERASE EOS, ¦ or
	ESC, J, PLUS	ERASE EOS, +

(Note 1) This character may be typed in as uppercase or lowercase character.

Figure B-4 (Part 1 of 2). Program Function Keys for the IBM 3101

Function	ASCII-CODE (-Sequence)	IBM 3101 Key
PFK 25	ESC, J, a	ERASE EOS, a (Note 1)
PFK 26	ESC, J, s	ERASE EOS, s (Note 1)
PFK 27	ESC, J, d	ERASE EOS, d (Note 1)
PFK 28	ESC, J, f	ERASE EOS, f (Note 1)
PFK 29	ESC, J, g	ERASE EOS, g (Note 1)
PFK 30	ESC, J, h	ERASE EOS, h (Note 1)
PFK 31	ESC, J, j	ERASE EOS, j (Note 1)
PFK 32	ESC, J, k	ERASE EOS, k (Note 1)
PFK 33	ESC, J, l	ERASE EOS, l (Note 1)
PFK 34	ESC, J, SEMI	ERASE EOS, ;
PFK 35	ESC, J, QUOTE	ERASE EOS, '
PFK 36	ESC, J, LBRACE or ESC, J, RBRACE	ERASE EOS, { or ERASE EOS, }
(Note 1) This character may be typed in as uppercase or lowercase character.		

Figure B-4 (Part 2 of 2). Program Function Keys for the IBM 3101

B.6.3 Setup Functions

The setup functions are introduced by pressing

```
ERASE EOS, "ACCENT" (pressed sequentially)
```

and then pressing one additional key, which produces the functions listed in Figure B-5 on page B-7.

Function	ASCII-CODE (-Sequence)	IBM 3101 Key
Set Column Tab	ESC, J, ACCENT, HT	→|
Delete Column Tab	ESC, J, ACCENT, DEL	DEL
Set Left Margin	ESC, J, ACCENT, CR	↵
Set Home Line	ESC, J, ACCENT, ESC, H	ALT and |←
Delete all Column Tabs, and reset Home Line and Left Margin	ESC, J, ACCENT, ESC, L	CLEAR
Improved Null Processing	ESC, J, ACCENT, N	N
3270 Null Processing	ESC, J, ACCENT, n	n
Zones Mode on	ESC, J, ACCENT, z	z
Zones Mode off	ESC, J, ACCENT, Z	Z
Reverse Enter/Newline Keys	ESC, J, ACCENT, e	e
Restore Enter/Newline Keys	ESC, J, ACCENT, E	E
Reverse Column and Field Tab Keys	ESC, J, ACCENT, c	c
Restore Column and Field Tab Keys	ESC, J, ACCENT, C	C
Alpha in Numeric-Only Field	ESC, J, ACCENT, v	v
3270 Numeric Fields	ESC, J, ACCENT, V	V
3278 Insert Mode	ESC, J, ACCENT, i	i
3277 Insert Mode	ESC, J, ACCENT, I	I
APL Mode on	ESC, J, ACCENT, a	a
APL Mode off	ESC, J, ACCENT, A	A
ASCII Input in APL Mode	ESC, J, ACCENT, m	m
Alternate Display of Attributes	ESC, J, ACCENT, d	d
Primary Display of Attributes	ESC, J, ACCENT, D	D
Suppress Pacing	ESC, J, ACCENT, P	P
Restore Pacing	ESC, J, ACCENT, p	p
Keyboard initiated Line Drop	ESC, J, ACCENT, PERIOD	.
Return to ENTER TERMINAL TYPE Msg.	ESC, J, ACCENT, COMMA	,
Alternate Keyboard Arrangement	ESC, J, ACCENT, q	q
Primary Keyboard Arrangement	ESC, J, ACCENT, Q	Q

Figure B-5. Setup Functions for the IBM 3101

B.6.4 Local Reset and Control Functions

There are six key sequences that are completely processed within the IBM 7171, and are not passed to the host. These sequences provide a way of controlling and clearing the data transmission to and from the terminal:

```
ALT and g               Master Reset
ALT and r               Character Error Reset
ALT and t               Keyboard Unlock
ALT and x               Type-ahead Purge
ALT and s               Pacing Start
ALT and q               Pacing Stop
```

B.6.5 Installation Hints

There are four 8-position switches 1, 2, 3 and 4 located on top of the keyboard under a cover.

Switches 1 to 3 should be set to match the following requirements: Full-duplex EIA RS-232-C Interface with 7 data bits, 1 stop bit, 1 parity bit testing on even parity (See Note below). This means that the following settings are required:

```
switch  1  2  3  4  5  6  7  8  position
   1    D  D  U  U  u  u  d  u  ( required:  U = Up, D = Down )
   2    U  U  u  d  d  d  d  d  ( suggested: u = up, d = down )
   3    u  d  u  u  d  d  d  d
```

Switch 4 selects the transmission rate (baud rate) of the terminal. Positions 1-4 select terminal baud rate, and positions 5-8 select printer baud rate. The switches should have one of the following baud rates (See Note below):

```
1/5 2/6 3/7 4/8    Baud Rate
 D   D   U   U        300          ( U = Up, D = Down )
 D   U   D   D        600
 D   U   D   U       1200
 D   U   U   U       2400
 U   D   D   D       4800
 U   D   D   U       9600
```

B.7 IBM Personal Computer Running in 3101 Emulation Mode

B.7.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding IBM PC Key Sequence Required
Enter	BS	↵
Clear	ESC, L	Home
Test Request	ESC, J, ESC, W	PgDn, Esc, W
Local Print	ESC, W	Esc, W
DUP Character	VT	Ctrl and k
Fieldmark Character	FF	Ctrl and l
Cursor Select	ESC, K	Ctrl and Home
Redisplay	SYN	Ctrl and v
Erase Input	ESC, J, ESC, K	PgDn, Ctrl and Home
Erase EOF	ESC, I	End
Delete Character	DEL	Del
Toggle Insert Mode	ESC, J, DEL	Ins
Field Tab	HT	→
Field Backtab	ESC, J, HT	←
Column Tab	ESC, J, ESC, C or ESC, HT	PgDn, → or ESC, →
Column Backtab	ESC, J, ESC, D or ESC, CR	PgDn, ← or Esc, ←
Indent	ESC, J, ESC, A	PgDn, ↑
Undent	ESC, J, ESC, B	PgDn, ↓
PA1	ESC, J, COMMA or ESC, J, LESS or ESC, J, z	Alt and F1 or PgDn, , or PgDn, < or PgDn, z (Note 1)
PA2	ESC, J, PERIOD or ESC, J, GREATER or ESC, J, x	Alt and F2 or PgDn, . or PgDn, > or PgDn, x (Note 1)
PA3	ESC, J, SLASH or ESC, J, QUESTION or ESC, J, c	Alt and F3 or PgDn, / or PgDn, ? or PgDn, c (Note 1)
Newline	CR	←
Home	ESC, H	Ctrl and PgUp
Cursor Up	ESC, A	↑
Cursor Down	ESC, B	↓
Cursor Right	ESC, C	→
Cursor Left	ESC, D	←

(Note 1) This character may be typed in as uppercase or lowercase character.

Figure B-6. Control and Cursor Movement Keys for the IBM Personal Computer running in 3101 Emulation Mode

B.7.2 Program Function Keys

Function	ASCII-CODE (-Sequence)	PC Key
PFK 1	ESC, J, 1	F1 or PgDn, 1
PFK 2	ESC, J, 2	F2 or PgDn, 2
PFK 3	ESC, J, 3	F3 or PgDn, 3
PFK 4	ESC, J, 4	F4 or PgDn, 4
PFK 5	ESC, J, 5	F5 or PgDn, 5
PFK 6	ESC, J, 6	F6 or PgDn, 6
PFK 7	ESC, J, 7	F7 or PgDn, 7
PFK 8	ESC, J, 8	F8 or PgDn, 8
PFK 9	ESC, J, 8	F9 or PgDn, 9
PFK 10	ESC, J, 9	F10 or PgDn, 0
PFK 11	ESC, J, HYPHEN	Shift and F1 or PgDn, -
PFK 12	ESC, J, EQUAL	Shift and F2 or PgDn, =
PFK 13	ESC, J, EXCLAIM or ESC, J, q	Shift and F3 or PgDn, ! or PgDn, q (Note 1)
PFK 14	ESC, J, AT or ESC, J, w	Shift and F4 or PgDn, @ or PgDn, w (Note 1)
PFK 15	ESC, J, POUND or ESC, J, e	Shift and F5 or PgDn, # or PgDn, e (Note 1)
PFK 16	ESC, J, DOLLAR or ESC, J, r	Shift and F6 or PgDn, $ or PgDn, r (Note 1)
PFK 17	ESC, J, PERCENT or ESC, J, t	Shift and F7 or PgDn, % or PgDn, t (Note 1)
PFK 18	ESC, J, UPARROW or ESC, J, y	Shift and F8 or PgDn, ˆ or PgDn, y (Note 1)
PFK 19	ESC, J, AND or ESC, J, u	Shift and F9 or PgDn, & or PgDn, u (Note 1)
PFK 20	ESC, J, STAR or ESC, J, i	Shift and F10 or PgDn, * or PgDn, i (Note 1)
PFK 21	ESC, J, LPAREN or ESC, J, o	Ctrl and F1 or PgDn, (or PgDn, o (Note 1)
PFK 22	ESC, J, RPAREN or ESC, J, p	Ctrl and F2 or PgDn,) or PgDn, p (Note 1)
PFK 23	ESC, J, LBRACK or ESC, J, RBRACK or ESC, J, UNDER	Ctrl and F3 or PgDn, [or PgDn,] or PgDn, _
PFK 24	ESC, J, BSLASH or ESC, J, BAR or ESC, J, PLUS	Ctrl and F4 or PgDn, \ or PgDn, ¦ or PgDn, +

(Note 1) This character may be typed in as uppercase or lowercase character.

Figure B-7 (Part 1 of 2). Program Function Keys for the IBM Personal Computer running in 3101 Emulation Mode

Function	ASCII-CODE (-Sequence)	PC Key
PFK 25	ESC, J, a	PgDn, a (Note 1)
PFK 26	ESC, J, s	PgDn, s (Note 1)
PFK 27	ESC, J, d	PgDn, d (Note 1)
PFK 28	ESC, J, f	PgDn, f (Note 1)
PFK 29	ESC, J, g	PgDn, g (Note 1)
PFK 30	ESC, J, h	PgDn, h (Note 1)
PFK 31	ESC, J, j	PgDn, j (Note 1)
PFK 32	ESC, J, k	PgDn, k (Note 1)
PFK 33	ESC, J, l	PgDn, l (Note 1)
PFK 34	ESC, J, SEMI	PgDn, ;
PFK 35	ESC, J, QUOTE	PgDn, '
PFK 36	ESC, J, LBRACE or ESC, J, RBRACE	PgDn, { or PgDn, }
(Note 1) This character may be typed in as uppercase or lowercase character.		

Figure B-7 (Part 2 of 2). Program Function Keys for the IBM Personal Computer running in 3101 Emulation Mode

B.7.3 Setup Functions

The setup functions are introduced by pressing

```
PgDn, "ACCENT" (pressed sequentially)
```

and then pressing one additional key, which produces the functions as described in Figure B-8 on page B-12.

Function	ASCII-CODE (-Sequence)	PC Key
Set Column Tab	ESC, J, ACCENT, HT	→|
Delete Column Tab	ESC, J, ACCENT, DEL	Del
Set left margin	ESC, J, ACCENT, CR	←
Set home line	ESC, J, ACCENT, ESC, H	Ctrl and PgUp
Delete all Column Tabs, and reset Home Line and Left Margin	ESC, J, ACCENT, ESC, L	Home
Improved Null Processing	ESC, J, ACCENT, N	N
3270 Null Processing	ESC, J, ACCENT, n	n
Zones Mode on	ESC, J, ACCENT, z	z
Zones Mode off	ESC, J, ACCENT, Z	Z
Reverse Enter/Newline Keys	ESC, J, ACCENT, e	e
Restore Enter/Newline Keys	ESC, J, ACCENT, E	E
Reverse Column and Field Tab Keys	ESC, J, ACCENT, c	c
Restore Column and Field Tab Keys	ESC, J, ACCENT, C	C
Alpha in Numeric-Only Field	ESC, J, ACCENT, v	v
3270 Numeric Fields	ESC, J, ACCENT, V	V
3278 Insert Mode	ESC, J, ACCENT, i	i
3277 Insert Mode	ESC, J, ACCENT, I	I
APL Mode on	ESC, J, ACCENT, a	a
APL Mode off	ESC, J, ACCENT, A	A
ASCII Input in APL Mode	ESC, J, ACCENT, m	m
Alternate Display of Attributes	ESC, J, ACCENT, d	d
Primary Display of Attributes	ESC, J, ACCENT, D	D
Suppress Pacing	ESC, J, ACCENT, P	P
Restore Pacing	ESC, J, ACCENT, p	p
Keyboard initiated Line Drop	ESC, J, ACCENT, PERIOD	.
Return to ENTER TERMINAL TYPE Msg.	ESC, J, ACCENT, COMMA	,
Alternate Keyboard Arrangement	ESC, J, ACCENT, q	q
Primary Keyboard Arrangement	ESC, J, ACCENT, Q	Q

Figure B-8. Setup Functions for the IBM Personal Computer running in 3101 Emulation Mode

B.7.4 Local Reset and Control Functions

There are six key sequences that are completely processed within the IBM 7171, and are not passed to the host. These sequences provide a way of controlling and clearing the data transmission to and from the terminal:

```
CTRL and g              Master Reset
CTRL and r              Character Error Reset
CTRL and t              Keyboard Unlock
CTRL and x              Type-ahead Purge
CTRL and s              Pacing Start
CTRL and q              Pacing Stop
```

B.7.5 Installation Hints

The IBM Personal Computer can be used with the IBM 7171 only when running a full screen terminal emulator program. Therefore, the IBM 3101 Emulation Program (IBM Program Number 6936712) using the IBM3101 table has to run first.

B.7.6 Starting the 3101 Emulation Program

Start the Emulation Program using the following steps:

1. Insert the Emulation Program working diskette in the diskette drive (drive A for a two-drive system).

2. Switch on the computer, or, if the IBM Personal Computer is already switched on, hold down the CTRL and ALT keys and at the same time press the DEL key.

3. When prompted, enter the current date in the form mm-dd-yy, where mm is the month, dd is the day, and yy is the last two digits of the year.

4. When prompted, enter the current time in the form hh:mm:ss.xx where hh is the hour, mm is the minutes, ss is the seconds, and xx is hundredths of a second (ss and xx are not required).

5. Type in the command "SETUP"

6. The Emulation Program begins and displays:

   IBM Personal Computer
   IBM 3101 Emulation Program
   Version 1.00 (C) Copyright IBM Corp. 1982
   Program name: Terminal Value Specification

   Function Selection Menu

   Choose:

   1. Select a specification file to run as a terminal
   2. Modify a specification file
   3. Create a specification file

4. Exit program

Type a number and press ENTER:

7. Now do one of the following:

   a. To use the sample YALEIUP.SET specification file, press '1' to get the following message:

   ```
   Enter filename (or press Enter to return
   to Function Selection Menu):
   ```

   b. To create a new specification file, press '3' to get the following prompt:

Line speed (Baud rate)	enter one of the following baud rates: 300, 600, 1200, 2400, 4800, or 9600
Block mode	enter "N" for Character mode
Half-Duplex?	enter "N" for Full-duplex
Parity?	enter "2" for even parity
Stop Bits?	enter "1" for one Stop bit
Automatic new line	enter "Y" or "N"
Automatic line feed	enter "Y" or "N"
Carriage return?	enter "Y" or "N"
Null suppress?	enter "Y" or "N"
Character sent at End of message?	enter "2" for CR
Scrolling?	enter "Y" or "N"
Prompt character from Host?	enter "0" for none
START/STOP (XON/XOFF) enabled?	enter "Y" or "N"

Note: Line speed and parity selection must conform with the specifications defined in the IBM 7171 Ports Area of memory for the appropriate communication line. Refer to section "4.4.2 Ports Area Layout" on page 4-24.

After the responses have been entered, give this set of specifications a filename for future reference. The next time the IBM 3101 Emulation Program is called, enter "1" on the above selection list and specify the filename to use the new set of specifications.

After the terminal specifications have been defined the following message appears:

```
Selection of terminal specifications complete.
```

8. Type in the command "TERMINAL"

9. The Emulation Program displays:

   IBM Personal Computer
   IBM 3101 Emulation Program
   Version 1.00 (C)Copyright IBM Corp. 1982
   Program name: Terminal Emulation

10. Wait a few seconds. When the copyright notice disappears the message appears:

```
***Establish connection if necessary***
```

At this point, if the line is a switched line, connection must be established. Then press the ENTER key and from now on the IBM Personal Computer will operate in 3101 emulation mode. Continue with the logon procedure.

B.8 TVI-912 and TVI-920 Terminal

The following tables apply for both TVI-912 and TVI-920 terminals.

B.8.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding TVI-912/920 Key Sequence Required
Enter	CR	RETURN (Note 1) or CTRL and m
Clear	SUB	CLEAR SPACE
Test Request	-	-
Local Print	-	-
DUP Character	-	-
Fieldmark Character	-	-
Cursor Select	-	-
Redisplay	SYN	Ctrl and v
Erase Input	-	-
Erase EOF	SOH, DEL, CR	FUNCT and RUBOUT
Delete Character	DEL	RUBOUT
Toggle Insert Mode	ESC	ESC
Field Tab	SOH, FF, CR	FUNCT and →
Field Backtab	SOH, BS, CR	FUNCT and ←
Column Tab	HT	TAB (Note 1)
Column Backtab	SOH, HT, CR	FUNCT and TAB (Note 1)
Indent	SOH, VT, CR	FUNCT and ↑
Undent	SOH, LF, CR	FUNCT and ↓
PA1	SOH, COMMA, CR or SOH, LESS, CR	FUNCT and , (Note 1) or FUNCT and <
PA2	SOH, PERIOD, CR or SOH, GREATER, CR	FUNCT and . (Note 1) or FUNCT and >
PA3	SOH, SLASH, CR or SOH, QUESTION, CR	FUNCT and / or FUNCT and ?
Newline	RS	HOME or CTRL and ~ or CTRL and ^
Home	SOH, RS, CR	FUNCT and HOME
Cursor Up	VT	↑ or CTRL and k
Cursor Down	LF	↓ or CTRL and j or LINE FEED
Cursor Right	FF	→ or CTRL and l
Cursor Left	BS	← or CTRL and h

(Note 1) This key can be found on the main keypad as well as on the additional numeric keypad on the right side of the TVI-912/920 keyboard.

Figure B-9. Control and Cursor Movement Keys for the Televideo TVI-912/920

B.8.2 Program Function Keys

Function	ASCII-CODE (-Sequence)	TVI-912/920 Key
PFK 1	SOH, 1, CR	FUNCT and 1 (Note 1)
PFK 2	SOH, 2, CR	FUNCT and 2 (Note 1)
PFK 3	SOH, 3, CR	FUNCT and 3 (Note 1)
PFK 4	SOH, 4, CR	FUNCT and 4 (Note 1)
PFK 5	SOH, 5, CR	FUNCT and 5 (Note 1)
PFK 6	SOH, 6, CR	FUNCT and 6 (Note 1)
PFK 7	SOH, 7, CR	FUNCT and 7 (Note 1)
PFK 8	SOH, 8, CR	FUNCT and 8 (Note 1)
PFK 9	SOH, 9, CR	FUNCT and 9 (Note 1)
PFK 10	SOH, 0, CR	FUNCT and 0 (Note 1)
PFK 11	SOH, HYPHEN, CR	FUNCT and -
PFK 12	SOH, EQUAL, CR	FUNCT and =
PFK 13	SOH, EXCLAIM, CR or	FUNCT and ! or
	SOH, q, CR	FUNCT and q (Note 2)
PFK 14	SOH, AT, CR or	FUNCT and @ or
	SOH, w, CR	FUNCT and w (Note 2)
PFK 15	SOH, POUND, CR or	FUNCT and # or
	SOH, e, CR	FUNCT and e (Note 2)
PFK 16	SOH, DOLLAR, CR or	FUNCT and $ or
	SOH, r, CR	FUNCT and r (Note 2)
PFK 17	SOH, PERCENT, CR or	FUNCT and % or
	SOH, t, CR	FUNCT and t (Note 2)
PFK 18	SOH, UPARROW, CR or	FUNCT and ^ or
	SOH, y, CR	FUNCT and y (Note 2)
PFK 19	SOH, AND, CR or	FUNCT and & or
	SOH, u, CR	FUNCT and u (Note 2)
PFK 20	SOH, ASTERISK, CR or	FUNCT and * or
	SOH, i, CR	FUNCT and i (Note 2)
PFK 21	SOH, LPAREN, CR or	FUNCT and (or
	SOH, o, CR	FUNCT and o (Note 2)
PFK 22	SOH, RPAREN, CR or	FUNCT and) or
	SOH, p, CR	FUNCT and p (Note 2)
PFK 23	SOH, LBRACK, CR or	FUNCT and [or
	SOH, RBRACK, CR or	FUNCT and] or
	SOH, UNDER, CR	FUNCT and _
PFK 24	SOH, BSLASH, CR or	FUNCT and \ or
	SOH, BAR, CR or	FUNCT and ¦ or
	SOH, PLUS, CR	FUNCT and +

(Note 1) This key can be found on the main keypad as well as on the additional numeric keypad on the right side of the TVI-912/920 keyboard.

(Note 2) This character may be typed in as uppercase or lowercase character.

Figure B-10 (Part 1 of 2). Program Function Keys for the Televideo TVI-912/920

Function	ASCII-CODE (-Sequence)	TVI-912/920 Key
PFK 25	SOH, a, CR	FUNCT and a (Note 1)
PFK 26	SOH, s, CR	FUNCT and s (Note 1)
PFK 27	SOH, d, CR	FUNCT and d (Note 1)
PFK 28	SOH, f, CR	FUNCT and f (Note 1)
PFK 29	SOH, g, CR	FUNCT and g (Note 1)
PFK 30	SOH, h, CR	FUNCT and h (Note 1)
PFK 31	SOH, j, CR	FUNCT and j (Note 1)
PFK 32	SOH, k, CR	FUNCT and k (Note 1)
PFK 33	SOH, l, CR	FUNCT and l (Note 1)
PFK 34	SOH, SEMI, CR	FUNCT and ;
PFK 35	SOH, QUOTE, CR	FUNCT and '
PFK 36	SOH, LBRACE, CR or SOH, RBRACE, CR	FUNCT and { or FUNCT and }
(Note 1) This character may be typed in as uppercase or lowercase character.		

Figure B-10 (Part 2 of 2). Program Function Keys for the Televideo TVI-912/920

B.8.3 Setup Functions

The setup functions are introduced by pressing

```
FUNCT and "ACCENT"   (pressed simultaneously)
```

and then pressing one additional key, which produces the functions as shown in Figure B-11 on page B-19.

Function	ASCII-CODE (-Sequence)	TVI-912/920 Key
Set Column Tab	SOH, ACCENT, CR, HT	TAB (Note 1)
Delete Column Tab	SOH, ACCENT, CR, DEL	RUBOUT
Set left margin	SOH, ACCENT, CR, RS	HOME
Set home line	SOH, ACCENT, CR, ESC	ESC
Delete all Column Tabs, and reset Home Line and Left Margin	SOH, ACCENT, CR, SUB	CLEAR SPACE
Improved Null Processing	SOH, ACCENT, CR, N	N
3270 Null Processing	SOH, ACCENT, CR, n	n
Zones Mode on	SOH, ACCENT, CR, z	z
Zones Mode off	SOH, ACCENT, CR, Z	Z
Reverse Enter/Newline Keys	SOH, ACCENT, CR, e	e
Restore Enter/Newline Keys	SOH, ACCENT, CR, E	E
Reverse Column and Field Tab Keys	SOH, ACCENT, CR, c	c
Restore Column and Field Tab Keys	SOH, ACCENT, CR, C	C
Alpha in Numeric-Only Field	SOH, ACCENT, CR, v	v
3270 Numeric Fields	SOH, ACCENT, CR, V	V
3278 Insert Mode	SOH, ACCENT, CR, i	i
3277 Insert Mode	SOH, ACCENT, CR, I	I
APL Mode on	SOH, ACCENT, CR, a	a
APL Mode off	SOH, ACCENT, CR, A	A
ASCII Input in APL Mode	SOH, ACCENT, CR, m	m
Alternate Display of Attributes	SOH, ACCENT, CR, d	d
Primary Display of Attributes	SOH, ACCENT, CR, D	D
Suppress Pacing	SOH, ACCENT, CR, P	P
Restore Pacing	SOH, ACCENT, CR, p	p
Keyboard initiated Line Drop	SOH, ACCENT, CR, PERIOD	.
Return to ENTER TERMINAL TYPE Msg.	SOH, ACCENT, CR, COMMA	,
Alternate Keyboard Arrangement	SOH, ACCENT, CR, q	q
Primary Keyboard Arrangement	SOH, ACCENT, CR, Q	Q
(Note 1) This key can be found on the main keypad as well as on the additional numeric keypad on the right side of the TVI-912/920 keyboard.		

Figure B-11. Setup Functions for the Televideo TVI-912/920

B.8.4 Local Reset and Control Functions

There are six key sequences that are completely processed within the IBM 7171, and are not passed to the host. These sequences provide a way of controlling and clearing the data transmission to and from the terminal:

```
CTRL and g          Master Reset
CTRL and r          Character Error Reset
CTRL and t          Keyboard Unlock
CTRL and x          Type-ahead Purge
CTRL and s          Pacing Start
CTRL and q          Pacing Stop
```

B.8.5 Installation Hints

There are two 10-position switches S1 and S2 located on the rear side of the Televideo TVI-912/920.

S1 selects the transmission rate (baud rate). Only one switch may be down at a time. One of the following switches must be down to select the baud rate indicated in the right column (See Note):

```
2   -   9600  baud
3   -   4800  baud
4   -   2400  baud
5   -   1200  baud
6   -    600  baud
7   -    300  baud
```

Do not select 19200 baud by setting switch 1. This will cause transmission errors.

Switch S2 specifies some functions. The switches must be set from left (1) to right (10) as follows: (U = Up, D = Down)

```
U D U D D D D U U D     (50 Hz Refresh)
```
or
```
U D U U D D D U U D     (60 Hz Refresh)
```

Meaning of switches:

```
 1  -  up   :    required
 2  - down  :    Standard Character Set
 3  -  up   :    full duplex
 4  - down  :    50 Hz  or
    -  up   :    60 Hz
 5  - down  :    send parity (See Note)
 6  - down  :    1 stop bit
 7  - down  :    7 data bits
 8  -  up   :    7 data bits
 9  -  up   :    even parity (See Note)
10  - down  :    required
```

Note: Line speed and parity selection must conform with the specifications defined in the IBM 7171 Ports Area of memory for the appropriate communication line. Refer to section "4.4.2 Ports Area Layout" on page 4-24.

B.9 TVI-950 Terminal

If the IBM 7171 sends the message: "enter terminal type:" to the TVI-950, two terminal types may be specified:

TVI950 The normal output is written to the screen using half intensity. The highlighted output is written to the screen using full intensity.

TVI950R The normal output is written to the screen using full intensity. The highlighted output is written to the screen using reverse video.

B.9.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding TVI-950 Key Sequence Required
Enter	CR	RETURN or ENTER or CTRL and m
Clear	SUB	CLEAR SPACE or CTRL and z
Test Request	ESC, 6 or ESC, ESC, 7	SHIFT and SEND or ESC, SEND
Local Print	ESC, ESC, E	ESC, LINE INSERT
DUP Character	ESC, ESC, Q	ESC, CHAR INSERT
Fieldmark Character	ESC, ESC, W	ESC, CHAR DELETE
Cursor Select	ESC, 7	SEND
Redisplay	ESC, ASTERISK	ESC and * or CTRL and g
Erase Input	ESC, ESC, R	ESC, LINE DELETE
Erase EOF	ESC, R	LINE DELETE
Delete Character	DEL	DEL
Toggle Insert Mode	ESC, E	LINE INSERT
Field Tab	ESC, T	LINE ERASE
Field Backtab	ESC, t or ESC, ESC, T	SHIFT and LINE ERASE or ESC, LINE ERASE
Column Tab	HT	TAB
Column Backtab	ESC, I	BACKTAB
Indent	ESC, Y	PAGE ERASE
Undent	ESC, y or ESC, ESC, Y	SHIFT and PAGE ERASE or ESC, PAGE ERASE

Figure B-12 (Part 1 of 2). Control and Cursor Movement Keys for the Televideo TVI-950

Function	Corresponding ASCII Code Character Sequence	Corresponding TVI-950 Key Sequence Required
PA1	SOH, COMMA, CR or SOH, LESS, CR	FUNCT and , (Note 1) or FUNCT and <
PA2	SOH, PERIOD, CR or SOH, GREATER, CR	FUNCT and . (Note 1) or FUNCT and >
PA3	SOH, SLASH, CR or SOH, QUESTION ,CR	FUNCT and / or FUNCT and ?
Newline	LF	LINE FEED or CTRL and j
Home	RS	HOME or CTRL and ^
Cursor Up	VT	↑ or CTRL and k
Cursor Down	SYN or LF	↓ or CTRL and v
Cursor Right	FF	→ or CTRL and l
Cursor Left	BS	← or CTRL and h or BACKSPACE
(Note 1) This key can be found on the main keypad as well as on the additional numeric keypad on the right side of the TVI-950 keyboard.		

Figure B-12 (Part 2 of 2). Control and Cursor Movement Keys for the Televideo TVI-950

B.9.2 Program Function Keys

Function	ASCII-CODE (-Sequence)	TVI-950 Key
PFK 1	SOH, AT, CR	F1 or FUNCT and @
PFK 2	SOH, A, CR	F2 or FUNCT and A
PFK 3	SOH, B, CR	F3 or FUNCT and B
PFK 4	SOH, C, CR	F4 or FUNCT and C
PFK 5	SOH, D, CR	F5 or FUNCT and D
PFK 6	SOH, E, CR	F6 or FUNCT and E
PFK 7	SOH, F, CR	F7 or FUNCT and F
PFK 8	SOH, G, CR	F8 or FUNCT and G
PFK 9	SOH, H, CR	F9 or FUNCT and H
PFK 10	SOH, I, CR	F10 or FUNCT and I
PFK 11	SOH, J, CR	F11 or FUNCT and J
PFK 12	ESC, Q	CHAR INSERT
PFK 13	SOH, ACCENT, CR	SHIFT and F1 or FUNCT and
PFK 14	SOH, a, CR	SHIFT and F2 or FUNCT and a
PFK 15	SOH, b, CR	SHIFT and F3 or FUNCT and b
PFK 16	SOH, c, CR	SHIFT and F4 or FUNCT and c
PFK 17	SOH, d, CR	SHIFT and F5 or FUNCT and d
PFK 18	SOH, e, CR	SHIFT and F6 or FUNCT and e
PFK 19	SOH, f, CR	SHIFT and F7 or FUNCT and f
PFK 20	SOH, g, CR	SHIFT and F8 or FUNCT and g
PFK 21	SOH, h, CR	SHIFT and F9 or FUNCT and h
PFK 22	SOH, i, CR	SHIFT and F10 or FUNCT and i
PFK 23	SOH, j, CR	SHIFT and F11 or FUNCT and j
PFK 24	ESC, q	SHIFT and CHAR INSERT

Figure B-13. Program Function Keys for the Televideo TVI-950

B.9.3 Setup Functions

The setup functions are introduced by pressing

`SHIFT and LINE DELETE  (pressed simultaneously)`

and then pressing one additional key, which will produce the functions as shown in Figure B-14 on page B-24.

Function	ASCII-CODE (-Sequence)	TVI-950 Key
Set Column Tab	ESC, O, HT	TAB
Delete Column Tab	ESC, O, DEL	DELETE
Set left margin	ESC, O, LF	LINE FEED
Set home line	ESC, O, RS	HOME
Delete all Column Tabs, and reset Home Line and Left Margin	ESC, O, SUB	CLEAR SPACE
Improved Null Processing	ESC, O, N	N
3270 Null Processing	ESC, O, n	n
Zones Mode on	ESC, O, z	z
Zones Mode off	ESC, O, Z	Z
Reverse Enter/Newline Keys	ESC, O, e	e
Restore Enter/Newline Keys	ESC, O, E	E
Reverse Column and Field Tab Keys	ESC, O, c	c
Restore Column and Field Tab Keys	ESC, O, C	C
Alpha in Numeric-Only Field	ESC, O, v	v
3270 Numeric Fields	ESC, O, V	V
3278 Insert Mode	ESC, O, i	i
3277 Insert Mode	ESC, O, I	I
APL Mode on	ESC, O, a	a
APL Mode off	ESC, O, A	A
ASCII Input in APL Mode	ESC, O, m	m
Alternate Display of Attributes	ESC, O, d	d
Primary Display of Attributes	ESC, O, D	D
Suppress Pacing	ESC, O, P	P
Restore Pacing	ESC, O, p	p
Keyboard initiated Line Drop	ESC, O, PERIOD	.
Return to ENTER TERMINAL TYPE Msg.	ESC, O, COMMA	,
Alternate Keyboard Arrangement	ESC, O, q	q
Primary Keyboard Arrangement	ESC, O, Q	Q

Figure B-14. Setup Functions for the Televideo TVI-950

Note: The ESC/LOC ESC-key on the TVI-950 keyboard is also used to invoke some local terminal display features (e.g. setting up a reverse background or special graphic symbols).

B.9.4 Local Reset and Control Functions

There are six key sequences that are completely processed within the IBM 7171, and are not passed to the host. These sequences provide a way of controlling and clearing the data transmission to and from the terminal:

```
CTRL and g          Master Reset
CTRL and r          Character Error Reset
CTRL and t          Keyboard Unlock
CTRL and x          Type-ahead Purge
CTRL and s          Pacing Start
CTRL and q          Pacing Stop
```

B.9.5 Installation Hints

Use the EIA RS-232-C Interface in full duplex mode with 7 data bits, 1 stop bit and 1 parity bit testing on even parity (See Note).

There are two 10-position switches S1 and S2 located on the rear side of the Televideo TVI-950.

S1 selects the transmission rate (baud rate), S2 selects the setting for parity. See Figure B-15 for appropriate setting of the switches and selection of baud rates.

Note: Line speed and parity selection must conform with the specifications defined in the IBM 7171 Ports Area of memory for the appropriate communication line. Refer to section "4.4.2 Ports Area Layout" on page 4-24.

	DIP SWITCH	POSITION (U=up,D=down)	FUNCTION
S1	1,2,3,4		Computer baud rate (see below)
	5	U	Seven-bit word structure
	6	D	One stop bit
	7,8,9,10		Printer baud rate (see below)

	DIP SWITCHES				
Terminal:	1	2	3	4	
Printer :	7	8	9	10	BAUD RATE
	D	U	U	D	300
	U	U	U	D	600
	D	D	D	U	1200
	D	U	D	U	2400
	D	D	U	U	4800
	D	U	U	U	9600
	U	U	U	U	19200

	DIP SWITCH	POSITION (U=up,D=down)	FUNCTION
S2	1	U	Duplex edit
	2	U	Blinking cursor
	3,4,5	D U U	Even parity(receive/transmit)
	6	U	Green on black display
	7,8	D U	Full duplex communication
	9	D or U	50 Hertz or 60 Hertz
	10	D or U	Keyclick on or off

Figure B-15. Dipswitch Setting for the Televideo TVI-950

B.10 ADM-31 Terminal

B.10.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding ADM-31 Key Sequence Required
Enter	CR	RETURN (Note 1) or CTRL and m
Clear	ESC, W	CHAR DELETE
Test Request	ESC, ESC, 5	ESC, SEND PAGE
Local Print	ESC, ESC, E	ESC, LINE INSERT
DUP Character	ESC, ESC, Q	ESC, CHAR INSERT
Fieldmark Character	ESC, ESC, W	ESC, CHAR DELETE
Cursor Select	ESC, 5	SEND PAGE
Redisplay	SYN	CTRL and v
Erase Input	ESC, ESC, R	ESC, LINE DELETE
Erase EOF	ESC, R	LINE DELETE
Delete Character	DEL	RUB
Toggle Insert Mode	ESC, E	LINE INSERT
Field Tab	ESC, T	LINE ERASE
Field Backtab	ESC, ESC, T	ESC, LINE ERASE
Column Tab	HT	TAB (Note 1)
Column Backtab	ESC, I	BACKTAB
Indent	ESC, Y	PAGE ERASE
Undent	ESC, ESC, Y	ESC, PAGE ERASE
PA1	SOH, COMMA, CR or SOH, LESS, CR	FUNCTION, , (Note 1) or FUNCTION, <
PA2	SOH, PERIOD, CR or SOH, GREATER, CR	FUNCTION, . (Note 1) or FUNCTION, >
PA3	SOH, SLASH, CR or SOH, QUESTION, CR	FUNCTION, / or FUNCTION, ?
Newline	US	PAGE NEWLINE
Home	RS	HOME or CTRL and ^
Cursor Up	VT	↑ or CTRL and k
Cursor Down	LF	↓ or CTRL and j or LINE FEED
Cursor Right	FF	→ or CTRL and l
Cursor Left	BS	← or CTRL and h

(Note 1) This key can be found on the main keypad as well as on the additional numeric keypad on the right side of the ADM-31 keyboard.

Figure B-16. Control and Cursor Movement Keys for the Lear Siegler ADM-31

B.10.2 Program Function Keys

Function	ASCII-CODE (-Sequence)	ADM-31 Key
PFK 1	SOH, 1, CR	FUNCTION, 1 (Note 1)
PFK 2	SOH, 2, CR	FUNCTION, 2 (Note 1)
PFK 3	SOH, 3, CR	FUNCTION, 3 (Note 1)
PFK 4	SOH, 4, CR	FUNCTION, 4 (Note 1)
PFK 5	SOH, 5, CR	FUNCTION, 5 (Note 1)
PFK 6	SOH, 6, CR	FUNCTION, 6 (Note 1)
PFK 7	SOH, 7, CR	FUNCTION, 7 (Note 1)
PFK 8	SOH, 8, CR	FUNCTION, 8 (Note 1)
PFK 9	SOH, 9, CR	FUNCTION, 9 (Note 1)
PFK 10	SOH, 0, CR	FUNCTION, 0 (Note 1)
PFK 11	SOH, HYPHEN, CR	FUNCTION, - (Note 1)
PFK 12	SOH, UPARROW, CR	FUNCTION, ^
PFK 13	SOH, EXCLAIM, CR or	FUNCTION, ! or
	SOH, q, CR	FUNCTION, q (Note 2)
PFK 14	SOH, DQUOTE, CR or	FUNCTION, " or
	SOH, w, CR	FUNCTION, w (Note 2)
PFK 15	SOH, POUND, CR or	FUNCTION, # or
	SOH, e, CR	FUNCTION, e (#)
PFK 16	SOH, DOLLAR, CR or	FUNCTION, $ or
	SOH, r, CR	FUNCTION, r (Note 2)
PFK 17	SOH, PERCENT, CR or	FUNCTION, % or
	SOH, t, CR	FUNCTION, t (Note 2)
PFK 18	SOH, AND, CR or	FUNCTION, & or
	SOH, y, CR	FUNCTION, y (Note 2)
PFK 19	SOH, ACCENT, CR or	FUNCTION, ` or
	SOH, u, CR	FUNCTION, u (Note 2)
PFK 20	SOH, LPAREN, CR or	FUNCTION, (or
	SOH, ASTERISK, CR or	FUNCTION, * or
	SOH, i, CR	FUNCTION, i (Note 2)
PFK 21	SOH, RPAREN, CR or	FUNCTION,) or
	SOH, o, CR	FUNCTION, o (Note 2)
PFK 22	SOH, p, CR	FUNCTION, p (Note 2)
PFK 23	SOH, AT, CR or	FUNCTION, @ or
	SOH, EQUAL, CR or	FUNCTION, = or
	SOH, UNDER, CR	FUNCTION, _

(Note 1) This key can be found on the main keypad as well as on the additional numeric keypad on the right side of the ADM-31 keyboard.

(Note 2) This character may be typed in as uppercase or lowercase character.

Figure B-17 (Part 1 of 2). Program Function Keys for the Lear Siegler ADM-31

Function	ASCII-CODE (-Sequence)	ADM-31 Key
PFK 24	SOH, LBRACK, CR or	FUNCTION, [or
	SOH, TILDE, CR or	FUNCTION, ~ or
	SOH, BSLASH, CR or	FUNCTION, \ or
	SOH, BAR, CR or	FUNCTION, ¦ or
	SOH, PLUS, CR	FUNCTION, +
PFK 25	SOH, a, CR	FUNCTION, a (Note 1)
PFK 26	SOH, s, CR	FUNCTION, s (Note 1)
PFK 27	SOH, d, CR	FUNCTION, d (Note 1)
PFK 28	SOH, f, CR	FUNCTION, f (Note 1)
PFK 29	SOH, g, CR	FUNCTION, g (Note 1)
PFK 30	SOH, h, CR	FUNCTION, h (Note 1)
PFK 31	SOH, j, CR	FUNCTION, j (Note 1)
PFK 32	SOH, k, CR	FUNCTION, k (Note 1)
PFK 33	SOH, l, CR	FUNCTION, l (Note 1)
PFK 34	SOH, SEMI, CR	FUNCTION, ;
PFK 35	SOH, COLON, CR or	FUNCTION, : or
	SOH, QUOTE, CR	FUNCTION, '
PFK 36	SOH, RBRACK, CR or	FUNCTION,] or
	SOH, LBRACE, CR or	FUNCTION, { or
	SOH, RBRACE, CR	FUNCTION, }

(Note 1) This character may be typed in as uppercase or lowercase character.

Figure B-17 (Part 2 of 2). Program Function Keys for the Lear Siegler ADM-31

B.10.3 Setup Functions

The setup functions are introduced by pressing

```
ESC, "ACCENT"   (pressed sequentially)
```

and then pressing one additional key, which produces the functions as shown in Figure B-18 on page B-30.

Function	ASCII-CODE (-Sequence)	ADM-31 Key
Set Column Tab	ESC, ACCENT, CR, HT	TAB (Note 1)
Delete Column Tab	ESC, ACCENT, CR, DEL	RUB
Set left margin	ESC, ACCENT, CR, US	PAGE NEWLINE
Set home line	ESC, ACCENT, CR, RS	HOME
Delete all Column Tabs, and reset Home Line and Left Margin	ESC, ACCENT, CR, ESC, W	CHAR DELETE
Improved Null Processing	ESC, ACCENT, CR, N	N
3270 Null Processing	ESC, ACCENT, CR, n	n
Zones Mode on	ESC, ACCENT, CR, z	z
Zones Mode off	ESC, ACCENT, CR, Z	Z
Reverse Enter/Newline Keys	ESC, ACCENT, CR, e	e
Restore Enter/Newline Keys	ESC, ACCENT, CR, E	E
Reverse Column and Field Tab Keys	ESC, ACCENT, CR, c	c
Restore Column and Field Tab Keys	ESC, ACCENT, CR, C	C
Alpha in Numeric-Only Field	ESC, ACCENT, CR, v	v
3270 Numeric Fields	ESC, ACCENT, CR, V	V
3278 Insert Mode	ESC, ACCENT, CR, i	i
3277 Insert Mode	ESC, ACCENT, CR, I	I
APL Mode on	ESC, ACCENT, CR, a	a
APL Mode off	ESC, ACCENT, CR, A	A
ASCII Input in APL Mode	ESC, ACCENT, CR, m	m
Alternate Display of Attributes	ESC, ACCENT, CR, d	d
Primary Display of Attributes	ESC, ACCENT, CR, D	D
Suppress Pacing	ESC, ACCENT, CR, P	P
Restore Pacing	ESC, ACCENT, CR, p	p
Keyboard initiated Line Drop	ESC, ACCENT, CR, PERIOD	.
Return to ENTER TERMINAL TYPE Msg.	ESC, ACCENT, CR, COMMA	,
Alternate Keyboard Arrangement	ESC, ACCENT, CR, q	q
Primary Keyboard Arrangement	ESC, ACCENT, CR, Q	Q
(Note 1) This key can be found on the main keypad as well as on the additional numeric keypad on the right side of the ADM-31 keyboard.		

Figure B-18. Setup Functions for the Lear Siegler ADM-31

B.10.4 Local Reset and Control Functions

There are six key sequences that are completely processed within the IBM 7171, and are not passed to the host. These sequences provide a way of controlling and clearing the data transmission to and from the terminal:

CTRL and g	Master Reset
CTRL and r	Character Error Reset
CTRL and t	Keyboard Unlock
CTRL and x	Type-ahead Purge
CTRL and s	Pacing Start
CTRL and q	Pacing Stop

B.10.5 Installation Hints

Use the EIA RS-232-C Interface in full duplex mode. There should be 7 data bits, 1 stop bit and 1 parity bit testing on even parity (See Note).

To set the proper baud rate there are two 16-position reels located on the rear side of the ADM-31.

Select one of the following baud rates for "Modem" and "Printer":

Reel Pos. No.		1	2	3	4	BAUD RATE
5	=	1	0	1	0	300
6	=	0	1	1	0	600
7	=	1	1	1	0	1200
10	=	0	1	0	1	2400
12	=	0	0	1	1	4800
14	=	0	1	1	1	9600

Note: Line speed and parity selection must conform with the specifications defined in the IBM 7171 Ports Area of memory for the appropriate communication line. Refer to section "4.4.2 Ports Area Layout" on page 4-24.

For further configuration control switch setting see the manufacturer-supplied User's Reference Manual.

B.11 ADM-3A Terminal

B.11.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding ADM-3A Key Sequence Required
Enter	CR	RETURN
Clear	SUB	CTRL and z
Test Request	-	-
Local Print	-	-
DUP Character	-	-
Fieldmark Character	-	-
Cursor Select	-	-
Redisplay	SYN	CTRL and v
Erase Input	-	-
Erase EOF	DLE	CTRL and p
Delete Character	DEL	RUB
Toggle Insert Mode	EM	CTRL and y
Field Tab	HT	CTRL and i
Field Backtab	SI	CTRL and o
Column Tab	ESC, HT	ESC, CTRL and i
Column Backtab	ESC, SI	ESC, CTRL and o
Indent	ESC, BS	ESC, CTRL and h
Undent	ESC, FF	ESC, CTRL and l
PA1	ESC, COMMA or	ESC, , or
	ESC, LESS	ESC, <
PA2	ESC, PERIOD or	ESC, . or
	ESC, GREATER	ESC, >
PA3	ESC, SLASH or	ESC, / or
	ESC, QUESTION	ESC, ?
Newline	NAK	CTRL and 5
Home	RS	CTRL and ^
Cursor Up	VT	CTRL and k or
		CTRL and +
Cursor Down	LF	CTRL and j or
		CTRL and * or
		LINE FEED
Cursor Right	FF	CTRL and l or
		CTRL and ,
Cursor Left	BS	CTRL and h or
		CTRL and (

Figure B-19. Control and Cursor Movement Keys for the Lear Siegler ADM-3A

B.11.2 Program Function Keys

Function	ASCII-CODE (-Sequence)	ADM-3A Key
PFK 1	ESC, 1	ESC, 1
PFK 2	ESC, 2	ESC, 2
PFK 3	ESC, 3	ESC, 3
PFK 4	ESC, 4	ESC, 4
PFK 5	ESC, 5	ESC, 5
PFK 6	ESC, 6	ESC, 6
PFK 7	ESC, 7	ESC, 7
PFK 8	ESC, 8	ESC, 8
PFK 9	ESC, 9	ESC, 9
PFK 10	ESC, 0	ESC, 0
PFK 11	ESC, HYPHEN	ESC, -
PFK 12	ESC, EQUAL	ESC, =
PFK 13	ESC, EXCLAIM or ESC, q	ESC, ! or ESC, q (Note 1)
PFK 14	ESC, AT or ESC, w	ESC, @ or ESC, w (Note 1)
PFK 15	ESC, POUND or ESC, e	ESC, # or ESC, e (Note 1)
PFK 16	ESC, DOLLAR or ESC, r	ESC, $ or ESC, r (Note 1)
PFK 17	ESC, PERCENT or ESC, t	ESC, % or ESC, t (Note 1)
PFK 18	ESC, UPARROW or ESC, y	ESC, ^ or ESC, y (Note 1)
PFK 19	ESC, AND or ESC, u	ESC, & or ESC, u (Note 1)
PFK 20	ESC, STAR or ESC, i	ESC, * or ESC, i (Note 1)
PFK 21	ESC, LPAREN or ESC, o	ESC, (or ESC, o (Note 1)
PFK 22	ESC, RPAREN or ESC, p	ESC,) or ESC, p (Note 1)
PFK 23	ESC, LBRACK or ESC, RBRACK or ESC, UNDER	ESC, [or ESC,] or ESC, _
PFK 24	ESC, BSLASH or ESC, BAR or ESC, PLUS	ESC, \ or ESC, ¦ or ESC, +

(Note 1) This character may be typed in as uppercase or lowercase character.

Figure B-20 (Part 1 of 2). Program Function Keys on the Lear Siegler ADM-3A

Function	ASCII-CODE (-Sequence)	ADM-3A Key
PFK 25	ESC, a	ESC, a (Note 1)
PFK 26	ESC, s	ESC, s (Note 1)
PFK 27	ESC, d	ESC, d (Note 1)
PFK 28	ESC, f	ESC, f (Note 1)
PFK 29	ESC, g	ESC, g (Note 1)
PFK 30	ESC, h	ESC, h (Note 1)
PFK 31	ESC, j	ESC, j (Note 1)
PFK 32	ESC, k	ESC, k (Note 1)
PFK 33	ESC, l	ESC, l (Note 1)
PFK 34	ESC, SEMI	ESC, ;
PFK 35	ESC, QUOTE	ESC, '
PFK 36	ESC, LBRACE or	ESC, { or
	ESC, RBRACE	ESC, }
(Note 1) This character may be typed in as uppercase or lowercase character.		

Figure B-20 (Part 2 of 2). Program Function Keys on the Lear Siegler ADM-3A

B.11.3 Setup Functions

The setup functions are introduced by pressing

```
ESC, "ACCENT" (pressed sequentially)
```

and then pressing one additional key, which produces the functions as shown in Figure B-21 on page B-35.

Function	ASCII-CODE (-Sequence)	ADM-3A Key
Set Column Tab	ESC, ACCENT, HT	CTRL and i
Delete Column Tab	ESC, ACCENT, DEL	RUB
Set left margin	ESC, ACCENT, NAK	CTRL and 5
Set home line	ESC, ACCENT, RS	CTRL and ^
Delete all Column Tabs, and reset Home Line and Left Margin	ESC, ACCENT, SUB	CTRL and z
Improved Null Processing	ESC, ACCENT, N	N
3270 Null Processing	ESC, ACCENT, n	n
Zones Mode on	ESC, ACCENT, z	z
Zones Mode off	ESC, ACCENT, Z	Z
Reverse Enter/Newline Keys	ESC, ACCENT, e	e
Restore Enter/Newline Keys	ESC, ACCENT, E	E
Reverse Column and Field Tab Keys	ESC, ACCENT, c	c
Restore Column and Field Tab Keys	ESC, ACCENT, C	C
Alpha in Numeric-Only Field	ESC, ACCENT, v	v
3270 Numeric Fields	ESC, ACCENT, V	V
3278 Insert Mode	ESC, ACCENT, i	i
3277 Insert Mode	ESC, ACCENT, I	I
APL Mode on	ESC, ACCENT, a	a
APL Mode off	ESC, ACCENT, A	A
ASCII Input in APL Mode	ESC, ACCENT, m	m
Alternate Display of Attributes	ESC, ACCENT, d	d
Primary Display of Attributes	ESC, ACCENT, D	D
Suppress Pacing	ESC, ACCENT, P	P
Restore Pacing	ESC, ACCENT, p	p
Keyboard initiated Line Drop	ESC, ACCENT, PERIOD	.
Return to ENTER TERMINAL TYPE Msg.	ESC, ACCENT, COMMA	,
Alternate Keyboard Arrangement	ESC, ACCENT, q	q
Primary Keyboard Arrangement	ESC, ACCENT, Q	Q

Figure B-21. Setup Functions for the Lear Siegler ADM-3A

B.11.4 Local Reset and Control Functions

There are six key sequences that are completely processed within the IBM 7171, and are not passed to the host. These sequences provide a way of controlling and clearing the data transmission to and from the terminal:

```
CTRL and g          Master Reset
CTRL and r          Character Error Reset
CTRL and t          Keyboard Unlock
CTRL and x          Type-ahead Purge
CTRL and s          Pacing Start
CTRL and q          Pacing Stop
```

B.11.5 Installation Hints

Use the EIA RS-232-C Interface in full duplex mode. There should be 7 data bits, 1 stop bit and 1 parity bit testing on even parity. The baud rate should be one of the following: 300, 600, 1200, 2400, 4800, 9600 or 19200.

Note: Line speed and parity selection must conform with the specifications defined in the IBM 7171 Ports Area of memory for the appropriate communication line. Refer to section "4.4.2 Ports Area Layout" on page 4-24.

B.12 VT-100 Terminal

B.12.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding VT-100 Key Sequence Required
Enter	CR	RETURN
Clear	ESC, O, M	ENTER (Note 1)
Test Request	-	-
Local Print	-	-
DUP Character	-	-
Fieldmark Character	-	-
Cursor Select	-	-
Redisplay	SYN	CTRL and v
Erase Input	-	-
Erase EOF	ESC, DEL	ESC, DELETE
Delete Character	DEL	DELETE
Toggle Insert Mode	ESC, O, n	. (PERIOD) (Note 1)
Field Tab	ESC, ESC, O, C	ESC, →
Field Backtab	ESC, ESC, O, D	ESC, ←
Column Tab	HT	TAB
Column Backtab	ESC, HT	ESC, TAB
Indent	ESC, ESC, O, A	ESC, ↑
Undent	ESC, ESC, O, B	ESC, ↓
PA1	ESC, COMMA or	ESC, , or
	ESC, LESS or	ESC, < or
	ESC, O, S	PF4
PA2	ESC, PERIOD or	ESC, . or
	ESC, GREATER or	ESC, > or
	ESC, O, m	- (HYPHEN) (Note 1)
PA3	ESC, SLASH or	ESC, / or
	ESC, QUESTION or	ESC, ? or
	ESC, O, l	, (COMMA) (Note 1)
Newline	LF	LINE FEED
Home	BS	BACKSPACE
Cursor Up	ESC, O, A	↑
Cursor Down	ESC, O, B	↓
Cursor Right	ESC, O, C	→
Cursor Left	ESC, O, D	←

(Note 1) This key can be found only on the additional numeric keypad on the right side of the VT-100 keyboard. Do not select a similar key on the main keypad!

Figure B-22. Control and Cursor Movement Keys for the VT-100

B.12.2 Program Function Keys

Function	ASCII-CODE (-Sequence)	VT100 Key
PFK 1	ESC, 1 or ESC, O, P	ESC, 1 (Note 2) or PF1
PFK 2	ESC, 2 or ESC, O, Q	ESC, 2 (Note 2) or PF2
PFK 3	ESC, 3 or ESC, O, R	ESC, 3 (Note 2) or PF3
PFK 4	ESC, 4 or ESC, O, w	ESC, 4 (Note 2) or 7 (Note 1)
PFK 5	ESC, 5 or ESC, O, x	ESC, 5 (Note 2) or 8 (Note 1)
PFK 6	ESC, 6 or ESC, O, y	ESC, 6 (Note 2) or 9 (Note 1)
PFK 7	ESC, 7 or ESC, O, t	ESC, 7 (Note 2) or 4 (Note 1)
PFK 8	ESC, 8 or ESC, O, u	ESC, 8 (Note 2) or 5 (Note 1)
PFK 9	ESC, 9 or ESC, O, v	ESC, 9 (Note 2) or 6 (Note 1)
PFK 10	ESC, 0 or ESC, O, q	ESC, 0 (Note 2) or 1 (Note 1)
PFK 11	ESC, HYPHEN or ESC, O, r	ESC, - (Note 2) or 2 (Note 1)
PFK 12	ESC, EQUAL or ESC, O, s	ESC, = or 3 (Note 1)
PFK 13	ESC, EXCLAIM or ESC, O, p, ESC, O, P or ESC, q	ESC, ! or 0 (Note 1), PF1 or ESC, q (Note 3)
PFK 14	ESC, AT or ESC, O, p, ESC, O, Q or ESC, w	ESC, @ or 0 (Note 1), PF2 or ESC, w (Note 3)
PFK 15	ESC, POUND or ESC, O, p, ESC, O, R or ESC, e	ESC, # or 0 (Note 1), PF3 or ESC, e (Note 3)

(Note 1) This key can be found <u>only</u> on the additional numeric keypad on the right side of of the VT-100. Do not select a similar key on the main keypad!

(Note 2) This key can be found <u>only</u> on the main keypad. Do not select a similar key on the additional numeric keypad on the right side of the VT-100 keyboard!

(Note 3) This character may be typed in as uppercase or lowercase character.

Figure B-23 (Part 1 of 3). Program Function Keys for the VT-100

Function	ASCII-CODE (-Sequence)	VT100 Key
PFK 16	ESC, DOLLAR or	ESC, $ or
	ESC, O, p, ESC, O, w or	0 (Note 1), 7 (Note 1) or
	ESC, r	ESC, r (Note 2)
PFK 17	ESC, PERCENT or	ESC, % or
	ESC, O, p, ESC, O, x or	0 (Note 1), 8 (Note 1) or
	ESC, t	ESC, t (Note 2)
PFK 18	ESC, UPARROW or	ESC, ^ or
	ESC, O, p, ESC, O, y or	0 (Note 1), 9 (Note 1) or
	ESC, y	ESC, y (Note 2)
PFK 19	ESC, AND or	ESC, & or
	ESC, O, p, ESC, O, t or	0 (Note 1), 4 (Note 1) or
	ESC, u	ESC, u (Note 2)
PFK 20	ESC, STAR or	ESC, * or
	ESC, O, p, ESC, O, u or	0 (Note 1), 5 (Note 1) or
	ESC, i	ESC, i (Note 2)
PFK 21	ESC, LPAREN or	ESC, (or
	ESC, O, p, ESC, O, v or	0 (Note 1), 6 (Note 1) or
	ESC, o	ESC, o
PFK 22	ESC, RPAREN or	ESC,) or
	ESC, O, p, ESC, O, q or	0 (Note 1), 1 (Note 1) or
	ESC, p	ESC, p (Note 2)
PFK 23	ESC, LBRACK or	ESC, [or
	ESC, LBRACE or	ESC, { or
	ESC, O, p, ESC, O, r or	0 (Note 1), 2 (Note 1) or
	ESC, UNDER	ESC, _
PFK 24	ESC, RBRACK or	ESC,] or
	ESC, RBRACE or	ESC, } or
	ESC, O, p, ESC, O, s or	0 (Note 1), 3 (Note 1) or
	ESC, BSLASH or	ESC, \ or
	ESC, BAR or	ESC, | or
	ESC, PLUS	ESC, +

(Note 1) This key can be found only on the additional numeric keypad on the right side of of the VT-100. Do not select a similar key on the main keypad!

(Note 2) This character may be typed in as uppercase or lowercase character.

Figure B-23 (Part 2 of 3). Program Function Keys for the VT-100

Function	ASCII-CODE (-Sequence)	VT-100 Key
PFK 25	ESC, ESC, O, P or ESC, a	ESC, PF1 or ESC, a (Note 2)
PFK 26	ESC, ESC, O, Q or ESC, s	ESC, PF2 or ESC, s (Note 2)
PFK 27	ESC, ESC, O, R or ESC, d	ESC, PF3 or ESC, d (Note 2)
PFK 28	ESC, ESC, O, w or ESC, f	ESC, 7 (*) or ESC, f (Note 2)
PFK 29	ESC, ESC, O, x or ESC, g	ESC, 8 (*) or ESC, g (Note 2)
PFK 30	ESC, ESC, O, y or ESC, h	ESC, 9 (*) or ESC, h (Note 2)
PFK 31	ESC, ESC, O, t or ESC, j	ESC, 4 (*) or ESC, j (Note 2)
PFK 32	ESC, ESC, O, u or ESC, k	ESC, 5 (*) or ESC, k (Note 2)
PFK 33	ESC, ESC, O, v or ESC, l	ESC, 6 (*) or ESC, l (Note 2)
PFK 34	ESC, ESC, O, q or ESC, SEMI	ESC, 1 (*) or ESC, ;
PFK 35	ESC, ESC, O, r or ESC, QUOTE	ESC, 2 (*) or ESC, '
PFK 36	ESC, ESC, O, s or ESC, CR	ESC, 3 (*) or ESC, RETURN

(Note 1) This key can be found only on the additional numeric keypad on the right side of of the VT-100. Do not select a similar key on the main keypad!

(Note 2) This character may be typed in as uppercase or lowercase character.

Figure B-23 (Part 3 of 3). Program Function Keys for the VT-100

B.12.3 Setup Functions

The setup functions are introduced by pressing

`ESC, "ACCENT" (pressed sequentially)`

and then pressing one additional key, which produces the functions listed in Figure B-24 on page B-41.

Function	ASCII-CODE (-Sequence)	VT-100 Key
Set Column Tab	ESC, ACCENT, HT	TAB
Delete Column Tab	ESC, ACCENT, DEL	DELETE
Set left margin	ESC, ACCENT, LF	LINE FEED
Set home line	ESC, ACCENT, BS	BACKSPACE
Delete all Column Tabs, and reset Home Line and Left Margin	ESC, ACCENT, ESC, O, M	ENTER (Note 1)
Improved Null Processing	ESC, ACCENT, N	N
3270 Null Processing	ESC, ACCENT, n	n
Zones Mode on	ESC, ACCENT, z	z
Zones Mode off	ESC, ACCENT, Z	Z
Reverse Enter/Newline Keys	ESC, ACCENT, e	e
Restore Enter/Newline Keys	ESC, ACCENT, E	E
Reverse Column and Field Tab Keys	ESC, ACCENT, c	c
Restore Column and Field Tab Keys	ESC, ACCENT, C	C
Alpha in Numeric-Only Field	ESC, ACCENT, v	v
3270 Numeric Fields	ESC, ACCENT, V	V
3278 Insert Mode	ESC, ACCENT, i	i
3277 Insert Mode	ESC, ACCENT, I	I
APL Mode on	ESC, ACCENT, a	a
APL Mode off	ESC, ACCENT, A	A
ASCII Input in APL Mode	ESC, ACCENT, m	m
Alternate Display of Attributes	ESC, ACCENT, d	d
Primary Display of Attributes	ESC, ACCENT, D	D
Suppress Pacing	ESC, ACCENT, P	P
Restore Pacing	ESC, ACCENT, p	p
Keyboard initiated Line Drop	ESC, ACCENT, PERIOD	.
Return to ENTER TERMINAL TYPE Msg.	ESC, ACCENT, COMMA	,
Alternate Keyboard Arrangement	ESC, ACCENT, q	q
Primary Keyboard Arrangement	ESC, ACCENT, Q	Q
(Note 1) This key can be found <u>only</u> on the additional numeric keypad on the right side of the VT-100 keyboard. Do not select a similar key on the main keypad.		

Figure B-24. Setup Functions for the VT-100

B.12.4 Local Reset and Control Functions

There are six key sequences that are completely processed within the IBM 7171, and are not passed to the host. These sequences provide a way of controlling and clearing the data transmission to and from the terminal:

```
CTRL and g              Master Reset
CTRL and r              Character Error Reset
CTRL and t              Keyboard Unlock
CTRL and x              Type-ahead Purge
CTRL and s              Pacing Start
CTRL and q              Pacing Stop
```

B.12.5 Installation Hints

Use the EIA RS-232-C Interface in full duplex mode (FDX-A) with 7 data bits, 1 stop bit and 1 parity bit testing on even parity. The baud rate should be one of the following: 300, 600, 1200, 2400, 4800, 9600 or 19200.

Note: Line speed and parity selection must conform with the specifications defined in the IBM 7171 Ports Area of memory for the appropriate communication line. Refer to section "4.4.2 Ports Area Layout" on page 4-24.

There is a built-in setup procedure to select these items in the VT-100. This procedure is described in the manufacturer-supplied *VT-100 Video Terminal User's Guide.*

Note: The VT-100 terminal connected to the IBM 7171 has no ability to switch the additional numeric keypad on the right side of the terminal between "Numeric Keypad Mode" and "Application Keypad Mode." The numeric keypad is always set to "Application Keypad Mode," because all keys on this pad are used for special purposes e.g. PA and PF keys. Therefore all digits (0 - 9), minus (-), comma (,) and period (.) symbols may only be typed in with keys on the main keypad.

B.13 DM-1520 and DM-1521 Terminal

The following tables apply for both DM-1520 and DM-1521 terminals.

B.13.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding DM-1520/21 Key Sequence Required
Enter	CR	RETURN
Clear	FF	CLEAR
Test Request	-	-
Local Print	-	-
DUP Character	-	-
Fieldmark Character	-	-
Cursor Select	-	-
Redisplay	SYN	CTRL and v
Erase Input	-	-
Erase EOF	VT	ERASE EOS
Delete Character	DEL	DEL
Toggle Insert Mode	GS	ERASE EOL
Field Tab	HT or	TAB or
	ESC, FS	ESC, →
Field Backtab	ESC, HT or	ESC, TAB or
	ESC, BS	ESC, ←
Column Tab	SO	PRINT
Column Backtab	ESC, SO	ESC, PRINT
Indent	ESC, US	ESC, ↑
Undent	ESC, LF	ESC, ↓
PA1	ESC, COMMA or	ESC, , or
	ESC, LESS	ESC, <
PA2	ESC, PERIOD or	ESC, . or
	ESC, GREATER	ESC, >
PA3	ESC, SLASH or	ESC, / or
	ESC, QUESTION	ESC, ?
Newline	SI	PRINT OFF
Home	EM	HOME
Cursor Up	US	↑
Cursor Down	LF	↓
Cursor Right	FS	→
Cursor Left	BS	←

Figure B-25. Control and Cursor Movement Keys for the DM-1520/21

B.13.2 Program Function Keys

Function	ASCII-CODE (-Sequence)	DM-1520/21 Key
PFK 1	ESC, 1	ESC, 1
PFK 2	ESC, 2	ESC, 2
PFK 3	ESC, 3	ESC, 3
PFK 4	ESC, 4	ESC, 4
PFK 5	ESC, 5	ESC, 5
PFK 6	ESC, 6	ESC, 6
PFK 7	ESC, 7	ESC, 7
PFK 8	ESC, 8	ESC, 8
PFK 9	ESC, 9	ESC, 9
PFK 10	ESC, 0	ESC, 0
PFK 11	ESC, HYPHEN	ESC, -
PFK 12	ESC, EQUAL	ESC, =
PFK 13	ESC, EXCLAIM or	ESC, ! or
	ESC, q	ESC, q (Note 1)
PFK 14	ESC, AT or	ESC, @ or
	ESC, w	ESC, w (Note 1)
PFK 15	ESC, POUND or	ESC, # or
	ESC, e	ESC, e (Note 1)
PFK 16	ESC, DOLLAR or	ESC, $ or
	ESC, r	ESC, r (Note 1)
PFK 17	ESC, PERCENT or	ESC, % or
	ESC, t	ESC, t (Note 1)
PFK 18	ESC, UPARROW or	ESC, ^ or
	ESC, y	ESC, y (Note 1)
PFK 19	ESC, AND or	ESC, & or
	ESC, u	ESC, u (Note 1)
PFK 20	ESC, STAR or	ESC, * or
	ESC, i	ESC, i (Note 1)
PFK 21	ESC, LPAREN or	ESC, (or
	ESC, o	ESC, o (Note 1)
PFK 22	ESC, RPAREN or	ESC,) or
	ESC, p	ESC, p (Note 1)
PFK 23	ESC, LBRACK or	ESC, [or
	ESC, RBRACK or	ESC,] or
	ESC, UNDER	ESC, _
PFK 24	ESC, BSLASH or	ESC, \ or
	ESC, BAR or	ESC, | or
	ESC, PLUS	ESC, +

(Note 1) This character may be typed in as uppercase or lowercase character.

Figure B-26 (Part 1 of 2). Program Function Keys for the DM-1520/21

Function	ASCII-CODE (-Sequence)	DM-1520/21 Key
PFK 25	ESC, a	ESC, a (Note 1)
PFK 26	ESC, s	ESC, s (Note 1)
PFK 27	ESC, d	ESC, d (Note 1)
PFK 28	ESC, f	ESC, f (Note 1)
PFK 29	ESC, g	ESC, g (Note 1)
PFK 30	ESC, h	ESC, h (Note 1)
PFK 31	ESC, j	ESC, j (Note 1)
PFK 32	ESC, k	ESC, k (Note 1)
PFK 33	ESC, l	ESC, l (Note 1)
PFK 34	ESC, SEMI	ESC, ;
PFK 35	ESC, QUOTE	ESC, '
PFK 36	ESC, LBRACE or	ESC, { or
	ESC, RBRACE	ESC, }
(Note 1) This character may be typed in as uppercase or lowercase character.		

Figure B-26 (Part 2 of 2). Program Function Keys for the DM-1520/21

B.13.3 Setup Functions

The setup functions are introduced by pressing

```
ESC, "ACCENT" (pressed sequentially)
```

and then pressing one additional key, which produces the functions listed in Figure B-27 on page B-46.

Function	ASCII-CODE (-Sequence)	DM-1520/21 Key
Set Column Tab	ESC, ACCENT, SO or	PRINT
	ESC, ACCENT, HT	TAB
Delete Column Tab	ESC, ACCENT, DEL	DEL
Set left margin	ESC, ACCENT, SI	PRINT OFF
Set home line	ESC, ACCENT, EM	HOME
Delete all Column Tabs, and reset Home Line and Left Margin	ESC, ACCENT, FF	CLEAR
Improved Null Processing	ESC, ACCENT, N	N
3270 Null Processing	ESC, ACCENT, n	n
Zones Mode on	ESC, ACCENT, z	z
Zones Mode off	ESC, ACCENT, Z	Z
Reverse Enter/Newline Keys	ESC, ACCENT, e	e
Restore Enter/Newline Keys	ESC, ACCENT, E	E
Reverse Column and Field Tab Keys	ESC, ACCENT, c	c
Restore Column and Field Tab Keys	ESC, ACCENT, C	C
Alpha in Numeric-Only Field	ESC, ACCENT, v	v
3270 Numeric Fields	ESC, ACCENT, V	V
3278 Insert Mode	ESC, ACCENT, i	i
3277 Insert Mode	ESC, ACCENT, I	I
APL Mode on	ESC, ACCENT, a	a
APL Mode off	ESC, ACCENT, A	A
ASCII Input in APL Mode	ESC, ACCENT, m	m
Alternate Display of Attributes	ESC, ACCENT, d	d
Primary Display of Attributes	ESC, ACCENT, D	D
Suppress Pacing	ESC, ACCENT, P	P
Restore Pacing	ESC, ACCENT, p	p
Keyboard initiated Line Drop	ESC, ACCENT, PERIOD	.
Return to ENTER TERMINAL TYPE Msg.	ESC, ACCENT, COMMA	,
Alternate Keyboard Arrangement	ESC, ACCENT, q	q
Primary Keyboard Arrangement	ESC, ACCENT, Q	Q

Figure B-27. Setup Functions for the DM-1520/21

B.13.4 Local Reset and Control Functions

There are six key sequences that are completely processed within the IBM 7171, and are not passed to the host. These sequences provide a way of controlling and clearing the data transmission to and from the terminal:

CTRL and g	Master Reset
CTRL and r	Character Error Reset
CTRL and t	Keyboard Unlock
CTRL and x	Type-ahead Purge
CTRL and s	Pacing Start
CTRL and q	Pacing Stop

B.13.5 Installation Hints

Use the EIA RS-232-C Interface in full duplex mode with 7 data bits, 1 stop bit and 1 parity bit testing on even parity. The baud rate should be one of the following: 300, 600, 1200, 2400, 4800, or 9600.

Note: Line speed and parity selection must conform with the specifications defined in the IBM 7171 Ports Area of memory for the appropriate communication line. Refer to section "4.4.2 Ports Area Layout" on page 4-24.

B.14 DM-3045 Terminal

B.14.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding DM-3045 Key Sequence Required
Enter	CR	RETURN
Clear	ESC, M	MR
Test Request	-	-
Local Print	-	-
DUP Character	-	-
Fieldmark Character	-	-
Cursor Select	-	-
Redisplay	SYN	CTRL and v
Erase Input	-	-
Erase EOF	ESC, J	SHIFT and ERASE EOS
Delete Character	DEL	RUB OUT
Toggle Insert Mode	ESC, P	INS CHAR
Field Tab	HT or ESC, ESC, C	TAB or ESC, →
Field Backtab	ESC, HT or ESC, BS	ESC, TAB or ESC, ←
Column Tab	ESC, V	ROLL
Column Backtab	ESC, ESC, V or ESC, W	ESC, ROLL or SHIFT and ROLL
Indent	ESC, ESC, A	ESC, ↑
Undent	ESC, LF	ESC, ↓
PA1	ESC, COMMA or ESC, LESS	ESC, , or ESC, <
PA2	ESC, PERIOD or ESC, GREATER	ESC, . or ESC, >
PA3	ESC, SLASH or ESC, QUESTION	ESC, / or ESC, ?
Newline	ESC, RBRACK	PRINT
Home	ESC, H	HOME
Cursor Up	ESC, A or ESC, Q	↑ or SHIFT and ↑
Cursor Down	LF or ESC, S	↓ or SHIFT and ↓
Cursor Right	ESC, C or ESC, R	→ or SHIFT and →
Cursor Left	BS or ESC, B	← or SHIFT and ←

Figure B-28. Control and Cursor Movement Keys for the DM-3045

B.14.2 Program Function Keys

Function	ASCII-CODE (-Sequence)	DM-3045 Key
PFK 1	ESC, 1 or ESC, p, CR	ESC, 1 or PF1
PFK 2	ESC, 2 or ESC, q, CR	ESC, 2 or PF2
PFK 3	ESC, 3 or ESC, r, CR	ESC, 3 or PF3
PFK 4	ESC, 4 or ESC, s, CR	ESC, 4 or PF4
PFK 5	ESC, 5 or ESC, t, CR	ESC, 5 or PF5
PFK 6	ESC, 6 or ESC, u, CR	ESC, 6 or PF6
PFK 7	ESC, 7 or ESC, v, CR	ESC, 7 or PF7
PFK 8	ESC, 8 or ESC, w, CR	ESC, 8 or PF8
PFK 9	ESC, 9 or ESC, x, CR	ESC, 9 or PF9
PFK 10	ESC, 0 or ESC, y, CR	ESC, 0 or PF10
PFK 11	ESC, HYPHEN or ESC, L, CR	ESC, - or XMIT LINE, RETURN
PFK 12	ESC, EQUAL or ESC, T, CR	ESC, = or XMIT, RETURN
PFK 13	ESC, EXCLAIM or ESC, ESC, p, CR	ESC, ! or ESC, PF1
PFK 14	ESC, AT or ESC, ESC, q, CR	ESC, @ or ESC, PF2
PFK 15	ESC, POUND or ESC, ESC, r, CR or ESC, e	ESC, # or ESC, PF3 or ESC, e (#)
PFK 16	ESC, DOLLAR or ESC, ESC, s, CR	ESC, $ or ESC, PF4
PFK 17	ESC, PERCENT or ESC, ESC, t, CR	ESC, % or ESC, PF5
PFK 18	ESC, UPARROW or ESC, ESC, u, CR or ESC, Y	ESC, ^ or ESC, PF6 or ESC, Y

(#) This character may be typed in as uppercase or lowercase character.

Figure B-29 (Part 1 of 2). Program Function Keys for the DM-3045

Function	ASCII-CODE (-Sequence)	DM-3045 Key
PFK 19	ESC, AND or	ESC, & or
	ESC, ESC, v, CR or	ESC, PF7 or
	ESC, U	ESC, U
PFK 20	ESC, STAR or	ESC, * or
	ESC, ESC, w, CR or	ESC, PF8 or
	ESC, i	ESC, i (Note 1)
PFK 21	ESC, LPAREN or	ESC, (or
	ESC, ESC, x, CR or	ESC, PF9 or
	ESC, o	ESC, o (Note 1)
PFK 22	ESC, RPAREN or	ESC,) or
	ESC, ESC, y, CR	ESC, PF10
PFK 23	ESC, LBRACK or	ESC, [or
	ESC, ESC, L, CR or	ESC, XMIT LINE, RETURN or
	ESC, UNDER	ESC, _
PFK 24	ESC, BSLASH or	ESC, \ or
	ESC, BAR or	ESC, | or
	ESC, ESC, T, CR or	ESC, XMIT, RETURN or
	ESC, PLUS	ESC, +
PFK 25	ESC, a	ESC, a
PFK 26	ESC, TILDE	ESC, ~
PFK 27	ESC, d	ESC, d (Note 1)
PFK 28	ESC, f	ESC, f (Note 1)
PFK 29	ESC, g	ESC, g (Note 1)
PFK 30	ESC, h	ESC, h
PFK 31	ESC, j	ESC, j
PFK 32	ESC, k	ESC, k (Note 1)
PFK 33	ESC, l	ESC, l
PFK 34	ESC, SEMI	ESC, ;
PFK 35	ESC, QUOTE	ESC, '
PFK 36	ESC, LBRACE	ESC, {

(Note 1) This character may be typed in as uppercase or lowercase character.

Figure B-29 (Part 2 of 2). Program Function Keys for the DM-3045

B.14.3 Setup Functions

The setup functions are introduced by pressing

```
ESC, "ACCENT" (pressed sequentially)
```

and then pressing one additional key, which produces the functions listed in Figure B-30 on page B-51.

Function	ASCII-CODE (-Sequence)	DM-3045 Key
Set Column Tab	ESC, ACCENT, HT	TAB
Delete Column Tab	ESC, ACCENT, DEL	RUB OUT
Set left margin	ESC, ACCENT, ESC, RBRACK	PRINT
Set home line	ESC, ACCENT, ESC, H	HOME
Delete all Column Tabs, and reset Home Line and Left Margin	ESC, ACCENT, ESC, M	MR
Improved Null Processing	ESC, ACCENT, N	N
3270 Null Processing	ESC, ACCENT, n	n
Zones Mode on	ESC, ACCENT, z	z
Zones Mode off	ESC, ACCENT, Z	Z
Reverse Enter/Newline Keys	ESC, ACCENT, e	e
Restore Enter/Newline Keys	ESC, ACCENT, E	E
Reverse Column and Field Tab Keys	ESC, ACCENT, c	c
Restore Column and Field Tab Keys	ESC, ACCENT, C	C
Alpha in Numeric-Only Field	ESC, ACCENT, v	v
3270 Numeric Fields	ESC, ACCENT, V	V
3278 Insert Mode	ESC, ACCENT, i	i
3277 Insert Mode	ESC, ACCENT, I	I
APL Mode on	ESC, ACCENT, a	a
APL Mode off	ESC, ACCENT, A	A
ASCII Input in APL Mode	ESC, ACCENT, m	m
Alternate Display of Attributes	ESC, ACCENT, d	d
Primary Display of Attributes	ESC, ACCENT, D	D
Suppress Pacing	ESC, ACCENT, P	P
Restore Pacing	ESC, ACCENT, p	p
Keyboard initiated Line Drop	ESC, ACCENT, PERIOD	.
Return to ENTER TERMINAL TYPE Msg.	ESC, ACCENT, COMMA	,
Alternate Keyboard Arrangement	ESC, ACCENT, q	q
Primary Keyboard Arrangement	ESC, ACCENT, Q	Q

Figure B-30. Setup Functions for the DM-3045

Note: The APL Setup functions may also be initiated by another key sequence:

APL Mode on	SO	CTRL and n
APL Mode off	ESC, RBRACE	ESC, "
ASCII Input in APL Mode	SI	CTRL and o

Do not type the Setup function introducer keys for these functions!

B.14.4 Local Reset and Control Functions

There are six key sequences that are completely processed within the IBM 7171, and are not passed to the host. These sequences provide a way of controlling and clearing the data transmission to and from the terminal:

CTRL and g	Master Reset
CTRL and r	Character Error Reset
CTRL and t	Keyboard Unlock
CTRL and x	Type-ahead Purge
CTRL and s	Pacing Start
CTRL and q	Pacing Stop

B.14.5 Installation Hints

Use the EIA RS-232-C interface in full duplex mode with 7 data bits, 1 stop bit and 1 parity bit testing on even parity. The baud rate should be one of the following: 300, 600, 1200, 2400, 4800, 9600 or 19200 (See Note).

In detail, the switches located under the screen should be as following:

Switch	Position	Description
REMOTE	in	Remote operating mode
DUPLEX	in	Full duplex mode
TAPE	out	Execute special characters
ASCII	in	Select ASCII characters
EXT CLOCK	out	unused
EIA	in	EIA RS-232-C interface

To set the appropriate baud rates, do the following:

1. Press: SHIFT and ALT MODE
2. Type: "L xxxx," where xxxx is one of the following baud rates: 300, 600, 1200, 2400, 4800 or 9600.
3. Press: RETURN

Note: Line speed and parity selection must conform with the specifications defined in the IBM 7171 Ports Area of memory for the appropriate communication line. Refer to section "4.4.2 Ports Area Layout" on page 4-24.

B.15 TYPETERM Typewriter Terminal

TYPETERM denotes a typical ASCII typewriter terminal, without specifying any particular brand or model. For this reason column 3 in each of the following tables, which normally shows the keys that initiate the function shown in column 1, has been left blank.

When it has been determined which keys on the ASCII terminal generate the ASCII code sequence shown in column 2, record them in column 3 for future reference.

B.15.1 Control and Cursor Movement Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding Key Sequence Required
Enter	CR	(See note at the start of this section)
Redisplay	SYN	
Delete Character	DEL	
Toggle Insert Mode	ESC, i or ESC, I	
Field Tab	ESC, t or ESC, T	
Field Backtab	ESC, b or ESC, B	
Column Tab	HT	
Column Backtab	ESC, HT	
PA1	ESC, COMMA or ESC, LESS	
PA2	ESC, PERIOD or ESC, GREATER	
PA3	ESC, SLASH or ESC, QUESTION	
Newline	LF	
Cursor Left	BS	

Figure B-31. Control and Cursor Movement Keys for a typical ASCII Typewriter Terminal

B.15.2 Program Function Keys

Function	Corresponding ASCII Code Character Sequence	Corresponding Key Sequence Required
PFK 1	ESC, 1	(See note at the start of this section)
PFK 2	ESC, 2	
PFK 3	ESC, 3	
PFK 4	ESC, 4	
PFK 5	ESC, 5	
PFK 6	ESC, 6	
PFK 7	ESC, 7	
PFK 8	ESC, 8	
PFK 9	ESC, 9	
PFK 10	ESC, 0	
PFK 11	ESC, HYPHEN	
PFK 12	ESC, EQUAL	
PFK 13	ESC, EXCLAIM or ESC, q	
PFK 14	ESC, AT or ESC, w	
PFK 15	ESC, POUND or ESC, e	
PFK 16	ESC, DOLLAR or ESC, r	
PFK 17	ESC, PERCENT	
PFK 18	ESC, UPARROW or ESC, u	
PFK 19	ESC, AND or ESC, u	
PFK 20	ESC, STAR	
PFK 21	ESC, LPAREN or ESC, o	
PFK 22	ESC, RPAREN or ESC, p	
PFK 23	ESC, UNDER or ESC, RBRACK or ESC, LBRACK	
PFK 24	ESC, PLUS or ESC, BSLASH or ESC, BAR	

Figure B-32 (Part 1 of 2). Program Function Keys for a typical ASCII Typewriter Terminal

Function	Corresponding ASCII Code Character Sequence	Corresponding Key Sequence Required
PFK 25	ESC, a	(See note at the start of this section)
PFK 26	ESC, s	
PFK 27	ESC, d	
PFK 28	ESC, f	
PFK 29	ESC, g	
PFK 30	ESC, h	
PFK 31	ESC, j	
PFK 32	ESC, k	
PFK 33	ESC, l	
PFK 34	ESC, SEMI	
PFK 35	ESC, QUOTE	
PFK 36	ESC, RBRACE or ESC, LBRACE	

Figure B-32 (Part 2 of 2). Program Function Keys for a typical ASCII Typewriter Terminal

Appendix C. IBM 7171 Support Utility for Modifying Terminal Tables

C.1 Brief Overview

In order for the the IBM 7171 to communicate with attached ASCII devices, the characteristics of each individual device and the line to which the device is attached must be known. If the device type is not currently supported, the user will need to supply the terminal parameters. The user will also have to define the line characteristics for each communication line(port) that is connected.

The Support Utility allows users to customize device types and to specify port configurations in a simple and easy to use manner. The supplied Support Utility allows the IBM 7171 system to be modified as user's requirements evolve. The Support Utility is found on the diskette labeled "IBM 7171 Support Utility" which is packaged with the IBM 7171. The Support Utility runs on an IBM personal computer. Data is maintained, edited, and stored locally on the IBM PC and then copied to the IBM 7171 via the asynchronous communications port on the PC.

Two different methods are provided for maintaining, terminal tables. One method, invoked by issuing the command 7171, provides a user friendly full-screen editing capability for generating terminal and port data. The second method, invoked by issuing the command 7171TML, provides a method of generating terminal tables using the a macro language specifically constrained to macro statements described in IBM Manual SB30-1911 *IBM Series/1 Yale ASCII Terminal Communication System II Program Description/Operations Manual* (including TNL SN60-1215 of 9/15/83). For details, refer to "C.12.2 IBM 7171 Support Utility Macro Language Versus Series/1 IUP" on page C-73.

C.2 Installation of the Support Utility

Following is a list of the minimum required hardware to run the supplied Support Utility.

- An IBM Personal computer system unit with a minimum of 128K memory. Any of the following PC family processors are sufficient: IBM PC, IBM PC XT, IBM PC XT/370, IBM PORTABLE PC, IBM PC AT.

- One 360Kb Diskette Drive.

- PC-DOS (2.0).

- Asynchronous communications feature (300-9600 baud, RS-232-C interface).
- One IBM 7171 direct attachment cable or modem cable (See *IBM 7171 ASCII Device Attachment Control Unit Description and Planning Guide,* GA24-4019 Appendix B, for a description of the cable wiring.)
- IBM 7171 Support Utility diskette.

Before running the Support Utility, make a copy of the entire diskette, and keep the original as a backup.

Following is a list of installation instructions:

1. Connect the IBM 7171 direct attachment cable or modem cable between the asynchronous communications port on the PC, and Port 0 of the IBM 7171. Port 0 is in the top right-hand corner of the communications ports on the IBM 7171.

   Note: All operations except I/0 can be performed without cable attachment.

2. Turn on IBM PC, and activate DOS.

3. Before any I/O between the PC and the IBM 7171 can take place, the communications port must be initialized through DOS. If I/O will not take place, continue with the next step. The settings must coincide with the port configuration set in the IBM 7171 for port 0. The initial setting is for auto baud detect, Even Parity, Seven Data Bits, One Stop Bit. To transmit at 9600 baud, therefore, the following DOS command must be issued first:

   ```
   MODE COM1:9600,E,7,1
   ```

   For more information, refer to " Dial-up" on page C-10.

4. Insert copy of IBM 7171 Support Utility diskette into diskette drive.

5. Issue the "7171" or "7171TML" command.

C.3 Configuring the IBM 7171 Ports

There are 5 steps involved in configuring the ports:

1. Get the current Ports status through the I/O function

2. Load the Ports data through Ports Load

3. Configure the Ports

4. Save the newly configured Ports

5. Transmit the new Ports Image file to the IBM 7171 through the I/O function.

C.4 Entering a New Terminal Table

C.4.1 Generating a Completely New Table

If the user is going to generate a new table, the following steps are involved:

1. Enter all of the data under the Terms Edit function:

   a. Features
   b. Input Sequences
   c. Output Sequences
   d. Login Sequence
   e. Reset Characters
   f. Graphic Characters.

2. Save the above information into a new .TRM file

3. Create a Link Control File

4. Link all desired terminal tables together

5. Transmit the new Terminal Image file to the IBM 7171.

C.4.2 Modifying an Existing Table

1. Load the terminal table either through the EXTRACT function if the terminal table came from the IBM 7171, or through the TERM LOAD function.

2. Modify all of the data under the Terms Edit function:

   a. Features
   b. Input Sequences
   c. Output Sequences
   d. Login Sequence
   e. Reset Characters

f. Graphic Characters.

3. Save the above information into a new .TRM file

4. Create a Link Control File

5. Link all desired terminal tables together

6. Transmit the new Terminal Image file to the IBM 7171. Note, transmission of the new Terminal Image file can be over dedicated or switchable communication lines.

C.5 IBM 7171 Support Utility

The Support Utility maintains its information in the form of a group of files. These files are stored on a diskette, and are useful only in a PC environment. There are 5 basic types of files:

- Terminal Data File (.TRM)
- Link Control File (.CTL)
- Terminal Image File (.IMG)
- Terminal TML File (.TML)
- Ports Image File (.IMG)

TERMINAL DATA is kept in files with a ".TRM" file extension. These files contain all of the device-specific information necessary for the IBM 7171 to communicate with a device.

Each .TRM file contains information about one specific device. The user must decide how many terminals to support. Once determined, the user must build an .IMG file that contains all of the information supplied in each .TRM file. The .TRM files are **LINKED** together into one **TERMINAL IMAGE** file. The creation of the .IMG file is governed by the **LINK CONTROL FILE** (file extension of .CTL) which directs the linking process. The .IMG file is transmitted to the IBM 7171.

TERMINAL IMAGE FILES contain the information for all of the user defined terminal tables. Since only one terminal table is edited at a time, a process exists to **EXTRACT** one terminal table from the entire .IMG file. The **EXTRACT** function is the opposite of the **LINK** function. The **EXTRACT** function separates out one terminal table from the entire **TERMINAL IMAGE FILE**, and creates a **TERMINAL DATA**, or .TRM file for the specific terminal type. The user is then free to edit the .TRM file for his unique requirements and re-link his .TRM files to produce a new **TERMINAL IMAGE FILE**

TERMINAL TML FILES are Macro Language source files specifically constrained tp macro statements described in IBM Manual SB30-1911 *IBM Series/1 Yale ASCII Terminal Communication System II Program Description/Operations Manual*(including TNL SN60-1215 of 9/15/83). These files are created using an editor such as EDLIN or the IBM Personal Editor. When all **TML** terminal tables are defined they are linked together into into a IBM 7171 loadable **.IMG** file using the **IBM 7171TML MLNK** function. As described above the **EXTRACT** can be used to select one **TERMINAL .TRM FILE** from this **.IMG** file. **.TRM** files created in this manner can be edited and re-linked using **7171.**

The **PORTS IMAGE FILE** contains all of the information necessary to describe the current status of the sixty-four ports. A **PORTS IMAGE FILE** may be loaded directly into memory, and then edited since all of the ports are edited at once.

The two **IMAGE** files (ports and terms) contain a byte-for-byte representation of NV-RAM. Only the **IMAGE** files (default extension of .IMG) may be transmitted to or received from the IBM 7171. All communication to the IBM 7171 is done via the **I/O** function.

C.6 Glossary

The following terms will be used in this Appendix, and appear on many of the Support Utility's menus.

Term	**Definition**
.CTL	Default file extension for control files. Control files are used to govern the process of linking several terminal files together into one image file.
.TML	Default file extension for files which contain IBM 7171 Macro Language commands for 7171MLNK processing.
.IMG	Default file extension for files which contain a hexadecimal "image" of IBM 7171 NV-RAM memory. This is the format that files must be in for communication to the IBM 7171.
.TRM	Default file extension for files which contain individual terminal table definitions. These files need to be linked together into an image file in order to be sent to the IBM 7171.
NV-RAM	This is the IBM 7171 memory area where all user supplied information is stored, 8K of Non-Volatile RAM.
BINKEY	A mode of input to the 7171 Support Utility. In this mode, ASCII characters are entered by depressing the key that is desired. Most keys can be entered this way. e.g. Pressing the Carriage Return key generates "CR", the BINKEY representation of the key depressed.
EDIT	The process of editing IBM 7171 terminal tables.
HEXKEY	A mode of input to the 7171 Support Utility where characters are represented by their hexadecimal values. Input is restricted to the hexadecimal characters, "0123456789ABCDEFabcdef."

C.7 IBM 7171 Support Utility

Following is a list of all of the Support Utility functions and data files that exist on the IBM 7171 Support Utility diskette:

Program Name	Description
7171 .COM	Main Menu Driver (Non-TML)
7171 .MNU	Menus for 7171
7171PORT .EXE	Ports Configuration Utility
7171TERM .EXE	Terminal Editing Utility
7171EDIT .EXE	Edit Utility for Control File
7171LINK .EXE	Linker Utility
7171IO .EXE	Input/Output to the IBM 7171
7171TML .COM	Main Menu Driver for TML processing
7171TML .MNU	Menus for 7171TML
7171MLNK .EXE	Link Utility for TML
7171MRPT .EXE	TML Report Generation
TST .CTL TST .TRM TSTOBJ .RPT TSTPORTS .RPT TSTSRC .RPT TSTTERMS .RPT	Sample Control File for Linking
ADM31 .TRM ADM3A .TRM DM1520 .TRM DM1521 .TRM DM3045 .TRM HARDCOPY .TRM IBM3101 .TRM PLOTTER .TRM TVI912 .TRM TVI920 .TRM TVI950 .TRM TVI950R .TRM TYPETERM .TRM VT100 .TRM	Terminal Data Files

Figure C-1. Contents of IBM 7171 Support Utility Diskette

C.8 IBM 7171 Support Utility Operation

There are two methods for using the IBM 7171 Support Utility. These are:

1. From full-screen Menus

2. From commands entered directly in DOS.

These two methods offer a user flexibility in performing support operations. Commands allow the invoking of functions without using the menu. These commands allow either direct execution from the DOS command line or incorporation into DOS bat files. Two figures Figure C-2 and Figure C-3 show the hierarchy of IBM 7171 support functions.

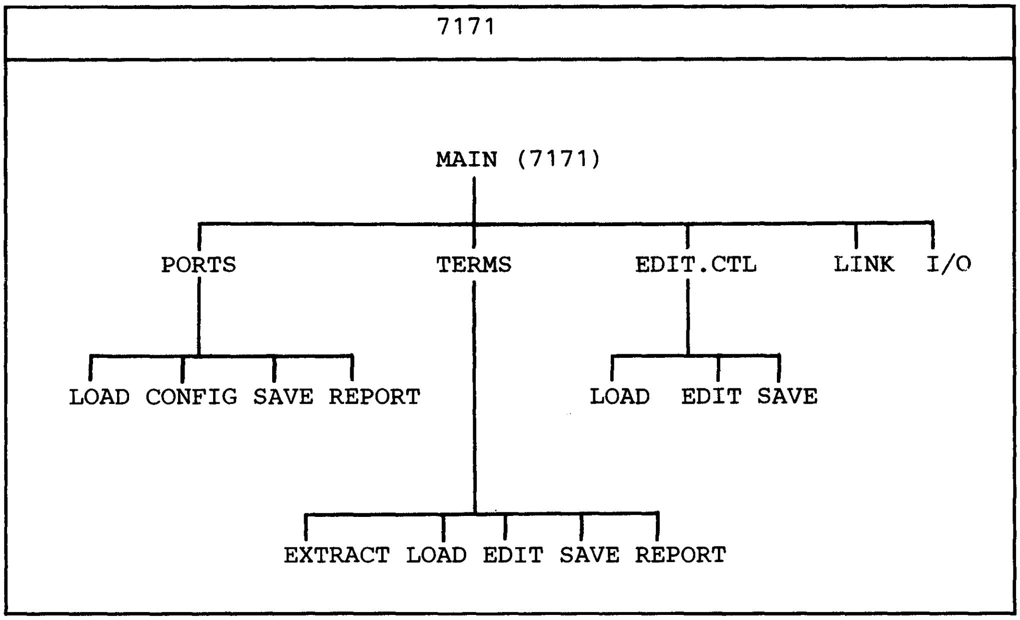


Figure C-2. Calling Tree of IBM 7171 Support Functions

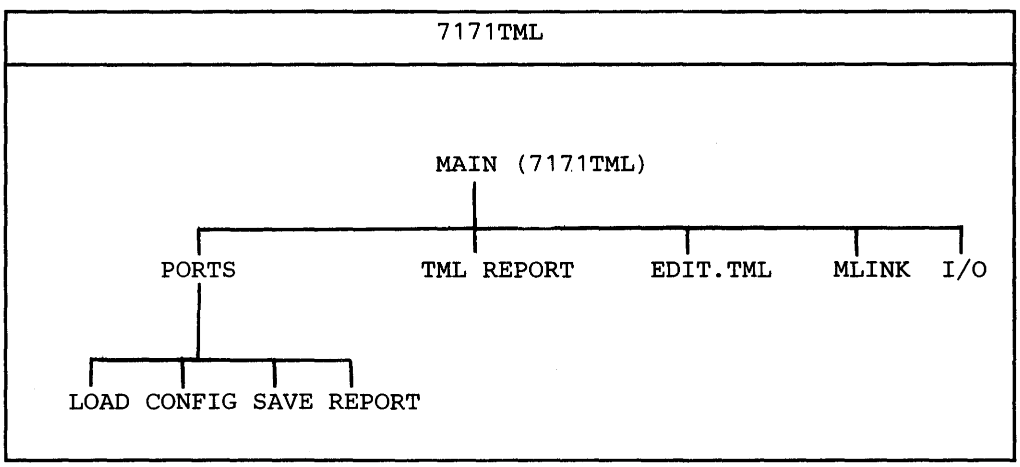


Figure C-3. Calling Tree of IBM 7171TML Support Functions

Here is a quick reference list for the nodes of the calling tree which will provide more details.

C.9 7171 Commands

C.9.1 Command Syntax Notation

We will use the following notation to indicate how the 7171 commands should be entered:

|	Items separated by a bar (|) mean that the user can choose one of the separated items.
{...}	Items in ({ }) are required.
[.....]	Items in square brackets ([ ]) are optional.

C.9.2 IBM 7171 Command Syntax

1. [x:]7171 [drive: [\path]]

   - Executing this command invokes the Menu Driven support for non-macro language table generation.

   - 'x' is the disk(ette) drive where 7171 resides.

   - 'drive' is where the utility programs which are called by 7171 reside, refer to the calling tree.

   - '\path' is the DOS pathname to the directory containing the utility programs and data files.

2. 7171PORT

   - Executing this command invokes the IBM 7171 Support Utility which performs all the functions necessary to configure ports.

3. 7171TERM

   - Executing this command invokes the IBM 7171 Support Utility which performs all the functions necessary to configure new terminals.

4. 7171LINK filename.CTL filename.IMG [>device]

   - Executing this command invokes the IBM 7171 Support Utility which performs the linking of all the .TRM files specified in the .CTL file.

   - >device redirects the output from the user's screen to the device specified. The device typically will be one of the following: PRN | filename.ext | CON

     If a device is not specified, the output will appear on the monitor.

5. 7171IO {Get | Put} {Ports | Terms} filename.IMG password [2]

- Executing this command invokes the IBM 7171 Support Utility which inputs or outputs .IMG files into or out of the IBM 7171.

- Either 'G' or 'P' must be entered in the place of Get or Put depending on the action desired.

- Either 'P' or 'T' must be entered in the place of Ports or Terms to specify the information which is to be moved.

- The optional 2 can be used to specify COM2 for communications instead of the default, which is COM1.

6. [x:]7171TML [drive: [\path]]

   - Executing this command invokes the Menu Driven support for TML table generation.

   - 'x' is the disk(ette) drive where 7171TML resides.

   - 'drive' is where the utility programs which are called by 7171TML reside, refer to the calling tree.

   - '\path' is the DOS pathname to the directory containing the utility programs and data files.

7. 7171MLNK filename.TML filename.IMG [>device]

   - Executing this command invokes the IBM 7171 Support Utility which performs the linking of all the .TRM files specified in the .TML file.

   - >device redirects the output from the user's screen to the device specified. The device typically will be one of the following: PRN | filename.ext | CON

     If a device is not specified, the output will appear on the monitor.

8. 7171MRPT

   - Executing this command invokes the IBM 7171 Support Utility which generates a TML Report. It presents a menu to the user for choosing the report form. Please note, this program will only unassemble .IMG files which where generated by 7171LINK.

Dial-up

The 7171IO program will allow transfer of .IMG files from an IBM PC over communication lines to the IBM 7171. For a dedicated line, an operator only has to invoke the 7171IO program via the menu selection process or by issuing the 7171IO command. If a switched line is used, first an operator must establish the connection over the switched network. The method of dialing into the system is modem dependent. Once the connection is established, then the operator proceeds as for a dedicated line. When using a dedicated line up to 9600 baud may be used. Baud rates of up to 1200 baud may be used over switched network lines.

As with all commands, this procedure may be implemented in a bat file.

A example of bat file for automatic dial up is:

```
ECHO  .................. >COM1
PAUSE
7171IO {Get | Put} {Ports | Term} filename.IMG password [2]
```

Executing the first command of the bat file outputs the dialing sequence to a modem which has auto-dialing capability. The "........" indicates that the user must put in a modem specific character string which will make the auto dial-in connection.

Execution of the PAUSE command will provide the prompt, Strike a key when ready. Press any key. The Pause command is not required. It is simply a method for delaying execution until the user is ready to continue.

Executing the last command invokes the IBM 7171 Support Utility which inputs or outputs .IMG files into or out of the IBM 7171. If 2 is not specified in the command the input or output will default to PC communications port COM1:.

C.10 IBM 7171 Support Utility from Menus

Figure C-4 illustrates a sample menu.

```
MENU TRACE                                           Clock
                          MENU NAME

   Input Area

   OPTIONS

   Comments



Information
Status
```

Figure C-4. Generic IBM 7171 Menu

The generic menu is divided into the following sections:

Menu Trace
: This area gives a trace of the menu currently active. It lists all of the menus that are necessary to go through in order to be at the current menu.

Clock
: The optional time and date function are printed here. These can be turned on or off by the use of PF keys. In particular PF5 turns the date on, PF6 turns the date off, PF7 turns on the time, and PF8 turns time off.

Menu Name
: This is the name of the current menu.

Input Area
: All user input is entered in this area.

Options
: All selections are made in this area. Selections might be: next menu, input field, or file name. The option is chosen by positioning the cursor over the option desired. The cursor may be positioned using any of the following: Right, Left, Up, and Down arrows, Home, End, Pg Up, and Pg Dn. These keys are located on the right hand side of the PC keyboard.

Comments

This field contains a one line description of the option currently indicated.

Information

This line contains useful information pertaining to the current menu, or state of the menu.

Status

Current input mode, and Insert, Caps lock, and Num Lock indicators are printed on this line

Note: From any menu or screen in the 7171 Support Utility, pressing CTRL-C or CTRL-BREAK will terminate the program and exit to DOS.

C.11 Main Menu

Figure C-5 illustrates the MAIN MENU that appears when 7171 is run.

```
-                                              mm/dd/yy  hh: mm AM
                        MAIN MENU

   PORTS  TERMS  EDIT .CTL   LINK  I/O  END

[F2] or ↵ to Select -  [F1] for Help
```

Figure C-5. Main Menu

There are 6 options on the main menu to select from:

PORTS
: This option allows the user to configure the communications ports on the IBM 7171. All ports processing is done after selecting this option.

TERMS
: All "Terminal" processing is done through this selection. This is device specific data that allows the IBM 7171 to communicate with that device.

EDIT CTL
: This is where the .CTL, or Link Control File is created.

LINK
: This is the function that builds an .IMG file out of the many .TRM files. The resultant .IMG file is then ready to transmit to the IBM 7171.

I/O
: All communication between the PC and the IBM 7171 is done through this function.

END
: This returns the user to DOS.

At any point, the user may press F1 for a function-key map, and a brief description of his alternatives.

One of the above options is selected by moving the cursor to the desired function name, and pressing either the Carriage Return key, or the PF2 key.

C.11.1 Ports Menu

Figure C-6 illustrates the Ports Menu.

```
- PORTS -
                         PORTS MENU

  LOAD   CONFIGURE   SAVE   REPORT   RETURN

[F2] or ↵ to Select - [F10] or [ESC] to Quit - [F1] for Help
```

Figure C-6. Ports Menu

The ports function will set the line characteristics for all of the 64 possible ports on the IBM 7171. If it is desired to change only one port's line characteristics, it would be better to use the Maintenance Facility Configure Port function. See "Chapter 9. Special Maintenance Facility and System Messages" on page 9-1 for details. The PORTS option in 7171 is useful to set all of the ports to a pre-defined state, or to generate reports for system documentation. The state of the ports is kept in a **Ports Image** file (default file extension is .IMG).

Within the Ports menu, the following functions are available:

LOAD
Load an existing .IMG file for ports configuration.

The LOAD function takes an .IMG from diskette and loads it into the PC memory for processing. The .IMG file may have been created by the PORTS SAVE command at an earlier time, or by I/O to the IBM 7171.

CONFIGURE
Edit current Ports data set.

The CONFIGURE option allows easy editing of the ports values that have been loaded into PC memory via the LOAD command. If no load has been done previously, the data is defaulted.

SAVE

Save current data set in an .IMG file

The SAVE function takes the ports data from PC memory, and creates a .IMG file containing all the current information. This must be done before transmitting the data to the IBM 7171 via the I/O routine. Any data that was altered during a CONFIGURE session will be stored as it exists at the time of the SAVE.

REPORT

Generate a hardcopy output of the Ports configuration.

RETURN

Return to Main Menu.

Ports Load

Figure C-7 illustrates the Ports Load screen.

```
- PORTS - LOAD
                     LOAD PORTS IMAGE FILE FOR CONFIGURATION

        Enter filename:

 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.IMG drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .IMG

[F2]=Proceed - [F10]=Cancel  -  Use  [Home] [End] to move cursor  -  [F1]=Help
Alphanumeric Input Mode
```

Figure C-7. Ports Load Screen

The user is being prompted for the name of the .IMG file that contains the PORTS information. For example, if an I/O was run and Ports information was stored in a file called PORTS.IMG, the input would be "PORTS," since the .IMG file extension is the default. If the .IMG file existed in a sub-directory, then the entire DOS path name would be required.

All of the standard PC editing keys may be used during file name entry.

Ports Configure

Figure C-8 Illustrates the Ports Configure screen.

```
 - PORTS - CONFIGURE
                           CONFIGURE IBM 7171 PORTS

  PORT #R   #C   BAUD   FLAG   TYPE  |    NOTES:
                                     |
   0   24   80     0    1A00         |     BAUD=0 => AUTO-BAUD
                                     |       = 300 600 1200 1800 2400 3600
   1   24   80     0    1A00         |         4800 9600 19200
                                     |       or (not auto-baud detectable)
   2   24   80     0    1A00         |    TYPE=<blank>=>End-user will be
                                     |         prompted for Terminal Type
   3   24   80     0    1A00         |    FLAG=(sum Flags selected)
                                     |       / 18xx= Even Parity  choose
   4   24   80     0    1A00         |        08xx= Odd  Parity /   one
                                     |     +    x4xx= 2 Stop bits(0=1 Stop)
   5   24   80     0    1A00         |     + / x3xx= 8 bit char   choose
                                     |        x2xx= 7 bit char  /   one
   6   24   80     0    1A00         |     + / xxx3= Local attach
                                     |        xxx0= Dialup or local attach
   7   24   80     0    1A00         |

 [F2] or ↵ to Select for Edit - [F10] or [ESC] to Return = [F1] for Help
```

Figure C-8. Ports Configure Screen

The following options may be set for each of the 64 ports:

Option	Argument
#R	The number of rows on the screen. 0 defaults to 24 rows
#C	Number of columns on the screen. 0 defaults to 80 columns.
BAUD	Baud rate. The following baud rates are supported by the IBM 7171 as auto baud detectable: 300, 600, 1200, 1800, 2400, 3600, 4800, 9600, 19200. If the baud rate is set to 0, then the terminal will default to auto baud detect mode. The following baud rates are supported, but are not detectable in auto baud mode: 50, 75, 110, 134, 150, 2000, 7200.
FLAG	Flag word. There are two bytes of flags. The meanings of these flags are as follows: Byte 1 Flags.

Bits	*Meaning*
10	Parity. 0 = Odd Parity, 1 = Even Parity
08	Parity Enable. 0 = Parity Disabled, 1 = Parity Enabled
04	Stop Bits. 0 = 1 stop bit, 1 = 2 stop bits
02-01	Data Bits. 00 = 5 data bits, 01 = 6 data bits, 10 = 7 data bits, 11 = 8 data bits.

Byte 2 Flags.

Bits	*Meaning*
02-01	Type of connection. 00 = Let TC default, 01 = Switched Network (Telephone Line), 10 = Leased Line, 11 = Direct Connect If both bits are 0, the TC will automatically determine the type of line that is attached.

TYPE This field specifies which terminal type to automatically assign to this port at line-connect time. This is the name of a valid terminal type whose Terminal Definition Table is currently stored in the IBM 7171. If this field is blank, then the terminal will prompt with the 'ENTER TERMINAL TYPE' message.

Note: 7171 will respond with an error message if there is not a Terminal Definition Table identified for the chosen terminal. To identify a user defined Terminal Definition Table, a **LINK** must be done, and the resulting .IMG file must be transmitted to the IBM 7171. Once this is accomplished, ports data must be retrieved from the IBM 7171. This is because the information necessary to identify a Terminal Definition Table is stored in the IBM 7171, and is collected when an I/O is done.

An option for a specific port is selected by first moving the cursor over the desired option, and then pressing either the F2 or the Carriage Return key. The value is then edited in the input area via the normal PC edit keys (i.e. insert, delete, cursor right, and cursor left). When the value has been modified, pressing F2 or the Carriage Return again will place the value into PC memory. When all the correct values have been set, the memory image is ready to be saved.

Note: Only eight ports are displayed at any one time. The rest of the ports may be accessed by using the "Pg Up" and "Pg Dn" keys located on the PC keypad.

Ports Save

Figure C-9 illustrates the ports save screen.

```
- PORTS - SAVE
                      SAVE CURRENT CONFIGURATION TO PORTS IMAGE FILE

      Enter filename:

 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.IMG drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .IMG

[F2]=Proceed - [F10]=Cancel - Use   [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-9. Ports Save Screen

Enter the file name of the .IMG file that the user wishes to save the current configuration in. This is the file name that will be used in the I/O function to transmit this configuration to the IBM 7171. If the user wishes, the file to be stored in another sub-directory, then the entire DOS path name must be entered.

Ports Report

Figure C-10 illustrates the ports report screen.

```
- PORTS - REPORT
                         PORTS REPORT TO PRINTER OR FILE

     Enter filename:

 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.RPT drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .RPT

 * PRN: filename will PRINT file - NOT SAVED ON DISK(ETTE)!

[F2]=Proceed - [F10]=Cancel - Use   [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-10. Ports Report Screen

This option will take the current ports configuration that is in memory, and generate output in a report form. The output may be sent to a file, and printed out later, or may be directed directly to a printer by entering "PRN:" as the file name.

Note: In order to generate a configuration report for a PORTS IMAGE file that exists on diskette, it is first necessary to load the .IMG file into memory using the PORTS LOAD option.

Figure C-11 illustrates the form of a typical ports report.

```
IBM 7171    (c) Copyright IBM Corp 1984                                          Page    1
                                                                                 07/04/84
IBM 7171 PORTS REPORT                                                             7:15 PM
```

PORT	#R	#C	BAUD	FLAG	TYPE	PORT	#R	#C	BAUD	FLAG	TYPE
0	24	80	0	1A00		32	24	80	0	1A00	
1	24	80	0	1A00		33	24	80	0	1A00	
2	24	80	0	1A00		34	24	80	0	1A00	
3	24	80	0	1A00		35	24	80	0	1A00	
4	24	80	0	1A00		36	24	80	0	1A00	
5	24	80	0	1A00		37	24	80	0	1A00	
6	24	80	0	1A00		38	24	80	0	1A00	
7	24	80	0	1A00		39	24	80	0	1A00	
8	24	80	0	1A00		40	24	80	0	1A00	
9	24	80	0	1A00		41	24	80	0	1A00	
10	24	80	0	1A00		42	24	80	0	1A00	
11	24	80	0	1A00		43	24	80	0	1A00	
12	24	80	0	1A00		44	24	80	0	1A00	
13	24	80	0	1A00		45	24	80	0	1A00	
14	24	80	0	1A00		46	24	80	0	1A00	
15	24	80	0	1A00		47	24	80	0	1A00	
16	24	80	0	1A00		48	24	80	0	1A00	
17	24	80	0	1A00		49	24	80	0	1A00	
18	24	80	0	1A00		50	24	80	0	1A00	
19	24	80	0	1A00		51	24	80	0	1A00	
20	24	80	0	1A00		52	24	80	0	1A00	
21	24	80	0	1A00		53	24	80	0	1A00	
22	24	80	0	1A00		54	24	80	0	1A00	
23	24	80	0	1A00		55	24	80	0	1A00	
24	24	80	0	1A00		56	24	80	0	1A00	
25	24	80	0	1A00		57	24	80	0	1A00	
26	24	80	0	1A00		58	24	80	0	1A00	
27	24	80	0	1A00		59	24	80	0	1A00	
28	24	80	0	1A00		60	24	80	0	1A00	
29	24	80	0	1A00		61	24	80	0	1A00	
30	24	80	0	1A00		62	24	80	0	1A00	
31	24	80	0	1A00		63	24	80	0	1A00	

Figure C-11. IBM 7171 Ports Report

C.11.2 Terms Menu

Figure C-12 illustrates the TERMS menu.

```
 - TERMS -
                              TERMS MENU

   EXTRACT  LOAD .TRM   EDIT   SAVE .TRM  REPORT  RETURN

[F2] or ↵ to Select - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-12. Terms Menu

The terms function is used to create or modify device specific .TRM files. Each of these .TRM files contains the information necessary for the IBM 7171 to completely support the use of that device. The .TRM files must later be linked together into one **TERMINAL IMAGE** file, default extension .IMG.

Within the Terms menu, the following options exist:

EXTRACT
: Extract terminal table(s) from a Terminal Image file.

  This is the function that removes one individual TERMINAL DATA, or .TRM, file from a TERMINAL IMAGE, or .IMG file.

LOAD .TRM
: Load Terminal table from .TRM file into PC memory.

  This function loads a **TERMINAL DATA** file into PC memory for editing.

EDIT
: Edit current terminal table that is stored in PC memory.

  This is the function that actually allows the user to modify current Terminal Data.

SAVE .TRM
Save current PC memory into a new .TRM file.

REPORT
Generate a report of the terminal table on the printer.

RETURN
Return to the main menu.

Terms Extract

Figure C-13 illustrates the LOAD TERMINAL IMAGE Screen for Terms Extract

```
- TERMS - EXTRACT
                         EXTRACT - LOAD TERMINAL IMAGE FILE

        Enter filename:

 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.IMG drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .IMG

[F2]=Proceed - [F10]=Cancel - Use [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-13. Load Terminal Image Screen

There are three steps to EXTRACTING a .TRM file from the TERMINAL IMAGE file:

1. Load the TERMINAL IMAGE file
2. Select the terminal to extract
3. Save the Extracted .TRM file.

The EXTRACT function can only generate .TRM files from user-specified terminal types. No native IBM 7171 terminal tables can be extracted using the EXTRACT function. Instead the native IBM 7171 terminals are supplied as .TRM files on the IBM 7171 Support Utility Diskette for user convenience. Thus, the EXTRACT function is only useful if a terminal table has been altered through the Operator console of the IBM 7171, or if the original .TRM file was destroyed.

This file must have been generated by 7171LINK. The .IMG file will be loaded into PC memory, and the user will then see the screen similar to the one illustrated in Figure C-14. The .IMG file may be gotten via the 7171IO program, however the .IMG file must still have been generated by the 7171LINK program.

Figure C-14 illustrates the EXTRACT-SELECT TERMINAL TYPE Screen.

```
- TERMS - EXTRACT
                         EXTRACT - SELECT TERMINAL

   HIGHLIGHTED TERM NAMES ARE IBM 7171 TERMS WHICH CANNOT BE EXTRACTED

ADM31    ADM3A    DM1520    DM1521    DM3045   IBM3101    TEST  TVI912

TVI920    TVI950    TVI950R    VT100    HARDCOPY    TYPETERM

IF ALL TERM NAMES ARE HIGHLIGHTED, [ESC] OR [F10] TO RETURN TO TERMS MENU
[F2] or ↵ to Select for Extracting - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-14. EXTRACT - Select Terminal Type Screen

To choose a terminal type to extract, place the cursor over the desired terminal name, and press F2 or Carriage Return. If the user chooses not to extract any terminal information, press F10 or ESC to return to the TERMS menu. Once a choice has been made, the following menu illustrated in Figure C-15 will be displayed.

Note: The file just extracted is put into PC memory. It is possible to go directly to the edit screen, if the user wishes to edit. The terminal data for the last extracted table will be displayed.

Figure C-15 illustrates the Extract-Save .TRM screen

```
- TERMS - EXTRACT
                      EXTRACT - SAVE EXTRACTED TERMINAL TO .TRM FILE

      Enter filename:

 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.TRM drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .TRM

[F2]=Proceed - [F10]=Return - Use [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-15. Extract - Save .TRM File

This will save the .TRM file that was extracted out of the .IMG file. Enter the file name, and press "F10" or Carriage Return. The user will then be returned to the SELECT screen for further choices. Pressing "ESC" or "F2" will return the user to the SELECT screen, saving nothing.

Load .TRM

Figure C-16 illustrates the LOAD TERMINAL DATA screen.

```
- TERMS - LOAD .TRM
                      LOAD .TRM FILE FROM DISK(ETTE)

   |     Enter filename:                                            |

 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.TRM drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .TRM




[F2]=Proceed - [F10]=Cancel - Use [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-16. Load Terminal Image File

Enter the file name of a TERMINAL DATA FILE. The file will be loaded into PC memory, and the terminal tables will be ready for editing.

Terminal Edit

Figure C-17 illustrates the EDIT menu.

```
- TERMS - EDIT -
                              EDIT MENU

 FEATURES  INPUT SEQ  OUTPUT SEQ  LOGIN SEQ  RESET CHARS  GRAPHIC  RETURN















[F2] or ↵ to Select - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-17. Terms Edit Menu

FEATURES
Edit Terminal Features Flags, Delay, and Cursor Origin.

INPUT SEQ
Edit Terminal Input Strings

OUTPUT SEQ
Edit Terminal Output Strings

LOGIN SEQ
Enter Terminal Initialization String

RESET CHARS
Enter Terminal Reset Characters

GRAPHIC
Enter Graphic Strings

RETURN
Return to TERMS menu

Please refer to "Chapter 4. Customizing IBM 7171 Tables" for a complete description of these terminal characteristics.

Terminal Features: Figure C-18 illustrates the Terminal Features Edit Screen.

```
- TERMS - EDIT - FEATURES
                    EDIT TERMINAL FEATURE FLAGS, DELAY, CURSOR ORIGIN

  DESCRIPTION              HEX

  Flags                    0000  (see FLAG BITS below)

  Delay                    0000  0 (milliseconds)

  Cursor Origin            00    0 (decimal)

  FLAG BITS:
               8000  Hardcopy Terminal
               4000  APL switchable
               1000  Bit paired terminal (used only with APL support)
               0800  Highlighting can be performed
               0400  Highlighting is a mode
               0080  Use NUL (X'00') instead of DEL (X'7F') for Pad
               0040  Can display lower right corner without scrolling
               0020  Use 'real' 3270 algorithm for nulls on READ MODIFIED
               0010  1: default pacing off; 0: default pacing one

[F2] or ↵ to Select for Edit - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-18. Terminal Features Edit Screen

To change any one of the values, place the cursor over the the appropriate field on the screen, and press <CR> or F2. Edit the value in the box, and press <CR> or F2 again to put a new value into PC memory.

Please note, the Cursor Origin defaults to X'00', zero. Binary Reposition Sequences often require a cursor Origin of SPACE, (X'20'). Also see "Output Sequence" on page C-33.

Input Sequence: Figure C-19 illustrates the Terminal Input Sequence Edit Screen.

```
- TERMS - EDIT - INPUT SEQ
                    EDIT INPUT SEQUENCES

  FUNCTION          HEX               BINKEY

  Clear

  Enter

  Home

  Left

  Right

  Up

  Down

  Field Tab
[F2] = Edit - [F9]=Dup Function - [F10] or [ESC]=Return - [F1] for Help
```

Figure C-19. Terminal Input Sequence Edit Screen

Two ways exist to edit the input sequences, HEX and BINKEY. To edit either field, first move the cursor to the appropriate space on the screen, and press either <CR> or F2.

In HEX input mode, enter the hex representation of the bytes desired, and press either <CR> or F2.

In BINKEY input, press the keys themselves that are desired. For example, if an input sequence is to be "ESC J <CR>," press the three keys, "ESC," "J," "<CR>." All ASCII control characters may be entered by pressing the "CTRL" key and a graphic character key such as "a," "b," etc, remember "CTRL-c" will terminate the program, use hex mode to input X'03'. To enter the string press F2. To delete a keystroke, use the "Del" key.

Pressing function key F9 will generate a new row entry on the screen. This entry will have the same function name as the function row on which the cursor is located. This allows generating more than one input sequence for the same function. A duplicate string may be erase by using the F2 in the normal editing mode and entering an empty sting.

Note: Only eight input sequences are displayed at any one time. The rest may be accessed by using the "Pg Up" and "Pg Dn" keys located on the PC keypad.

Output Sequence: Figure C-20 illustrates the Terminal Output Sequence Edit.

```
- TERMS - EDIT - OUTPUT SEQ
                     EDIT OUTPUT SEQUENCES

  OUTPUT CSS          HEX  /  BINKEY on line after HEX
  Reposition          FF
                      n/a
  Erase EOL           FF
                      n/a
  Local Print         FF
                      n/a
  Tone                FF
                      n/a
  Cursor Left         FF
                      n/a
  Cursor Right        FF
                      n/a
  Cursor Up           FF
                      n/a
  Cursor Down         FF
                      n/a

[F2] or ↵ to Select for Edit - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-20. Output Sequences

Two ways exist to edit the output sequences, HEX and BINKEY. To edit either field, first move the cursor to the appropriate space on the screen, and press either <CR> or F2.

In HEX input mode, enter the hexadecimal value of the characters desired, and press either <CR> or F2.

In BINKEY input, press the keys themselves that are desired. For example, if an output sequence is to be "ESC J <CR>," press the three keys, "ESC," "J," "<CR>." To enter the string into memory, press F2.

Note: Only eight output sequences are displayed at any one time. The rest may be accessed by using the "Pg Up" and "Pg Dn" keys located on the PC keypad.

Note: In the special case of the reposition string, the following bytes must be entered in HEX mode. Their equivalent BINKEY display is also shown:

Byte	BINKEY representation
XF4	u_y (USERY)
XF6	u_x (USERX)
XF8	c_y (CHARY)
XFA	c_x (CHARX)

XFC b_y (BINY)

XFE b_x (BINX)

Also see "Terminal Features" on page C-31 for Cursor Origin details.

Login Sequence: Figure C-21 illustrates the Terminal Login Sequence Edit.

```
- TERMS - EDIT - LOGIN SEQ
                       EDIT LOGIN INITIALIZATION SEQUENCE
-------------------------------------------------------------------------------
FF

-------------------------------------------------------------------------------
n/a

-------------------------------------------------------------------------------
[F2] or ↵ to Select for Edit - [F10] or [ESC] to Return - [F1] for Help
NOTE: IN-PLACE EDIT, MAX LENGTH = 240 CHARS
```

Figure C-21. Login Initialization Sequence

The Terminal Initialization String is edited separately because it can be up to 240 characters in length, whereas the other output strings can only be 16 bytes in length.

Two ways exist to edit the login sequences, HEX and BINKEY. To edit either field, first move the cursor to the appropriate space on the screen, and press either <CR> or F2. In HEX input mode, enter the representation of the bytes desired, and press either <CR> or F2.

In BINKEY input, press the keys themselves that are desired. For example, if an initialization string is to be "ESC J <CR>," press the three keys, "ESC," "J," "<CR>." To enter the string into memory, press F2.

Reset Characters: Figure C-22 illustrates the reset character screen.

```
- TERMS - EDIT - RESET CHARS
                     EDIT RESET CHARACTERS

 CHARACTER               HEX   BINKEY  |   NOTES:
                                       |
 Reset Introducer ....   FF      n/a   |   ANY UNUSED CHARACTERS SHOULD BE
                                       |   X'FF' NOT X'00' = null character
 Master Reset ........   FF      n/a   |        ---
                                       |
 Char Error Reset ....   FF      n/a   |
                                       |
 3270 'Reset' ........   FF      n/a   |
                                       |
 Input Buffer Flush ..   FF      n/a   |
                                       |
 Pacing Start Char ...   FF      n/a   |
                                       |
 Pacing Stop Char ....   FF      n/a   |
                                       |
 Maintenance Facility.   FF      n/a   |

[F2] or ↵ to Select for Edit - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-22. Reset Characters

Two ways exist to edit the reset characters, HEX and BINKEY. To edit either field, first move the cursor to the appropriate space on the screen, and press either <CR> or F2. In HEX input mode, enter the representation of the bytes desired, and press either <CR> or F2.

In BINKEY input, press the keys themselves that are desired. For example, if an reset character is to be "CTRL - G," press the key, "CTRL - G." To enter the string into memory, press F2.

Graphics Strings Figure C-23 illustrates the graphic string screen.

```
- TERMS - EDIT - GRAPHIC
               EDIT GRAPHIC RENDITION CONTROL STRING

 DESCRIPTION             HEX          BINKEY

 Primary                 0000000000   NUL NUL NUL NUL NUL
                         --  --  --   0-- 1-- 2-- 3-- 4--
 Alternate               0000000000   NUL NUL NUL NUL NUL
                         --  --  --   0-- 1-- 2-- 3-- 4--
                         0 1 2 3 4    (See Notes below)

 NOTES:

 0 = Non-display
 1 = Normal display unprotected
 2 = High Intensity unprotected
 3 = Normal display protected
 4 = High Intensity protected

[F2] or ↵ to Select for Edit - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-23. Set Graphics Strings

Two ways exist to edit the graphics strings, HEX and BINKEY. To edit either field, first move the cursor to the appropriate space on the screen, and press either <CR> or F2. In HEX input mode, enter the representation of the bytes desired, and press either <CR> or F2.

In BINKEY input, press the keys themselves that are desired. For example, if an graphics byte is to be "CTRL - G," press the key, "CTRL - G." To enter the string into memory, press F2.

Save .TRM File

```
- TERMS - SAVE .TRM
                  SAVE .TRM FILE TO DISK(ETTE)

    Enter 8 character Terminal name:

[F2]=Proceed - [F10]=Cancel - Use [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-24. Save Terminal Name

Two pieces of information are required to save a terminal data file:

1. Name of the Terminal Type

2. Name of Terminal Data File

The name of the terminal type will be the name the user types in at the "ENTER TERMINAL TYPE" message. The name of the file is the name that all of the data will be stored under. This name is also used in the control file at link time.

The first screen that appears when a save is initiated is illustrated in Figure C-24. After the terminal name is entered, press F2 or Carriage Return to continue on to the next screen, illustrated in Figure C-25.

```
- TERMS - SAVE .TRM
                     SAVE .TRM FILE TO DISK(ETTE)

     Enter filename:

 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.TRM drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .TRM

[F2]=Proceed - [F10]=Cancel - Use [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-25. Save .TRM File

This is the screen on which the name of the .TRM file is entered. After the name is entered, press F2 or Carriage Return to proceed, F10 or ESC will return the user back to the TERMS menu.

Terms Report

Figure C-26 illustrates the terms report screen.

```
- TERMS - REPORT
                TERMINAL REPORT TO PRINTER OR FILE
    ┌──────────────────────────────────────────────────────────────────┐
    │     Enter filename:                                              │
    └──────────────────────────────────────────────────────────────────┘
 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.RPT drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .RPT

 * PRN: filename will PRINT file - NOT SAVED ON DISK(ETTE)!

[F2]=Proceed - [F10]=Cancel - Use  [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-26. Terms Report Screen

This option will take the current terminal configuration that is in memory, and generate output in a report form. The output may be sent to a file, and printed out later, or may be directed directly to a printer by entering "PRN:" as the file name.

Note: In order to generate a configuration report for a TERMS IMAGE file that exists on diskette, it is first necessary to load the .IMG file into memory using the TERMS LOAD option. Figure C-27 illustrates a typical Terminal Report.

```
IBM 7171 (c) Copyright IBM Corp 1984                                        Page    1
                                                                            07/06/84
'TEST' TERMINAL TABLE DEFINITION                                            5:01   PM

FEATURES:

 DESCRIPTION        HEX

 Flags              0800
 Delay              0000  0 (milliseconds)
 Cursor Origin      01    1 (decimal)

RESET CHARACTERS (RCHRS):

 CHARACTER              HEX BINKEY

 Reset Introducer .... FF  n/a
 Master Reset ........ 12  DC2
 Char Error Reset .... FF  n/a
 3270 'Reset' ........ FF  n/a
 Input Buffer Flush .. FF  n/a
 Pacing Start Char ... 11  XON
 Pacing Stop Char .... 13  XOFF
 Maintenance Facility. 0F  SI

GRAPHIC RENDITION CONTROL CHARACTERS (SGR):

 DESCRIPTION       HEX          BINKEY

 Primary           02 00 01 00 07  STX NUL SOH NUL BEL
 Alternate         02 01 00 01 07  STX SOH NUL SOH BEL
                   0- 1- 2- 3- 4-  0-- 1-- 2-- 3-- 4--

OUTPUT SEQ (CSS):

 OUTPUT CSS     HEX  /  BINKEY on line after HEX

 Reposition     1B5BF83BFA48
                ESC [   c_y ;   c_x H
 Erase EOL      1B5B4B
                ESC [   K
 Local Print    1B5B50
                ESC [   P
 Tone           07
                BEL
 Cursor Left    1B5B44
                ESC [   D
 Cursor Right   1B5B43
                ESC [   C
 Cursor Up      1B5B41
                ESC [   A
 Cursor Down    1B5B42
                ESC [   B
 Enter Insert   1B5B77496E736572740D
                ESC [   w   I   n   s   e   r   t   CR
 Exit Insert    1B5B770D
                ESC [   w   CR
 Disconnect     1B5B481B5B4A
                ESC [   H   ESC [   J
 Clear          1B5B481B5B4A
                ESC [   H   ESC [   J
```

Figure C-27 (Part 1 of 3). Terminal Report

```
IBM 7171     (c) Copyright IBM Corp 1984                        Page    2
                                                                07/06/84
'TEST' TERMINAL TABLE DEFINITION                                5:01   PM

OUTPUT SEQ (CSS):

 OUTPUT CSS     HEX  /  BINKEY on line after HEX

 Login Init     1B5B481B5B4A1B5B771B5B376D204C4F474F4E200D
                ESC [    H    ESC [    J    ESC [    w    ESC [    7    m    spc L    O
                G    O    N    spc CR
 Illegal Char   3A
                :
 APL Illegal    <none>
 APL Chars ON   <none>
 APL Chars OFF  <none>
 Display Mode   1B5BEC6D
                ESC [    c_f m
 <Reserved>     <none>

INPUT SEQ (NODE):

 INPUT SEQ      HEX                  BINKEY

 Clear          1B5B4A               ESC [    J
 Enter          0D                   CR
 Home           1B5B48               ESC [    H
 Left           1B5B44               ESC [    D
 Right          1B5B43               ESC [    C
 Up             1B5B41               ESC [    A
 Down           1B5B42               ESC [    B
 Tab            09                   HT
 Tab Back       1B09                 ESC HT
 New Line       0A                   LF
 Insert         1B5B3468             ESC [    4    h
 Delete         7F                   DEL
 Erase EOF      1B5B4E               ESC [    N
 Insert         1B5B346C             ESC [    4    l
 Erase Input    1B5B4B               ESC [    K
 DUP            04                   EOT
 FM             06                   ACK
 Sys Request    1B535953             ESC S    Y    S
 PA 1           1B504131             ESC P    A    1
 PA 2           1B504132             ESC P    A    2
 PFK 01         1B3031               ESC 0    1
 Reshow         1B52                 ESC R
 Col Tab        14                   DC4
 Col Tab Back   02                   STX
 Set Tab        1B54                 ESC T
 Set Margin     1B4D                 ESC M
 Delete Tab     1B44                 ESC D
 Set Home       1B48                 ESC H
 Indent         1B49                 ESC I
 Undent         1B55                 ESC U
 Disconnect     18                   CAN
 Re-connect     0C                   FF
 Zones Off      1B5A                 ESC Z
 Zones On       1A                   SUB
 Set Num only   0E                   SO
 Set Alpha/Num  01                   SOH
 3277 INSERT    1B3737               ESC 7    7
```

Figure C-27 (Part 2 of 3). Terminal Report

```
IBM 7171    (c) Copyright IBM Corp 1984                                   Page   3
                                                                          07/06/84
'TEST' TERMINAL TABLE DEFINITION                                          5:01  PM

INPUT SEQ (NODE):

 INPUT SEQ      HEX                    BINKEY

 3278 INSERT    1B3738                 ESC 7   8
 Local Print    1B5B50                 ESC [   P
 Cursor Select  1B43                   ESC C
 PFK 02         1B3032                 ESC 0   2
 PFK 03         1B3033                 ESC 0   3
 PFK 04         1B3034                 ESC 0   4
 PFK 05         1B3035                 ESC 0   5
 PFK 06         1B3036                 ESC 0   6
 PFK 07         1B3037                 ESC 0   7
 PFK 08         1B3038                 ESC 0   8
 PFK 09         1B3039                 ESC 0   9
 PFK 10         1B3130                 ESC 1   0
 PFK 11         1B3131                 ESC 1   1
 PFK 12         1B3132                 ESC 1   2
 PFK 13         1B3133                 ESC 1   3
 PFK 14         1B3134                 ESC 1   4
 PFK 15         1B3135                 ESC 1   5
 PFK 16         1B3136                 ESC 1   6
 PFK 17         1B3137                 ESC 1   7
 PFK 18         1B3138                 ESC 1   8
 PFK 19         1B3139                 ESC 1   9
 PFK 20         1B3230                 ESC 2   0
 PFK 21         1B3231                 ESC 2   1
 PFK 22         1B3232                 ESC 2   2
 PFK 23         1B3233                 ESC 2   3
 PFK 24         1B3234                 ESC 2   4
 PFK 25         1B3235                 ESC 2   5
 PFK 26         1B3236                 ESC 2   6
 PFK 27         1B3237                 ESC 2   7
 PFK 28         1B3238                 ESC 2   8
 PFK 29         1B3239                 ESC 2   9
 PFK 30         1B3330                 ESC 3   0
 PFK 31         1B3331                 ESC 3   1
 PFK 32         1B3332                 ESC 3   2
 PFK 33         1B3333                 ESC 3   3
 PFK 34         1B3334                 ESC 3   4
 PFK 35         1B3335                 ESC 3   5
 PFK 36         1B3336                 ESC 3   6

 ================
```

Figure C-27 (Part 3 of 3). Terminal Report

C.11.3 Edit .CTL Menu

Figure C-28 illustrates the EDIT .CTL menu.

```
- EDIT  -
                        EDIT .CTL MENU

   LOAD .CTL  EDIT .CTL  SAVE .CTL  RETURN

[F2] or ↵ to Select - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-28. Edit .CTL Menu

The .CTL function allows a user to create or modify a link control file. The .CTL file is in a basic format, and can also be created with a standard text editor. The file format is a string of arguments, separated by spaces or commas. The arguments specify which .TRM files are to be linked into the TERMINAL IMAGE file that is to be transmitted to the IBM 7171, and in what order the terminal names which appear after the "VALID TYPES ARE" message are in. The arguments are:

7171:native_termname
This tells the linker to include this native IBM 7171 terminal type into the list at this point. Valid terminal names are: ADM31, ADM3A, DM1520, DM1521, DM3045, IBM3101, TVI912, TVI920, TVI950, TVI950R, VT100, HARDCOPY, and TYPETERM.

file.trm
This is the name of a .TRM TERMINAL DATA FILE. The terminal name that was specified in the menu illustrated in Figure C-24 is added to the list of terminal names at this point.

+
Indicates that a Carriage Return is to be put out after the previous terminal name in the message.

-
Indicates that the rest of the terminal names are to be hidden.

Warning: Only those terminals whose names are listed in the .CTL file, either native IBM 7171 or user generated, will be allowable on the IBM 7171. If the user wishes all of the native terminal names to be included, then the user MUST list the terminal names in the .CTL file.

Within the Edit .CTL menu, the following functions are available:

LOAD
Load an existing .CTL file for editing. The LOAD function takes a .CTL from diskette and loads it into the PC memory for processing.

EDIT
Edit current .CTL data set. The EDIT option allows easy editing of the .CTL file that has been loaded into PC memory via the LOAD command. If no load has been done previously, the screen will appear blank, and a new file may be created.

SAVE
Save current data set to a .CTL file. The SAVE function takes the current information from PC memory, and creates a .CTL file. This must be done before linking the .TRM files together.

RETURN
Return to Main Menu.

Load .CTL

Figure C-29 illustrates the LOAD .CTL screen.

```
- EDIT - LOAD .CTL
                      LOAD .CTL CONTROL FILE FOR EDITING

        Enter filename:

 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.CTL drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .CTL

[F2]=Proceed - [F10]=Cancel - Use [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-29. Load .CTL File

Enter the file name of a Control File. The file will be loaded into PC memory, and be ready for editing.

Edit .CTL

Figure C-30 illustrates the edit screen.

```
EDIT - EDIT .CTL
                        EDIT .CTL CONTROL FILE




[F2] or ↵ to Select - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-30. Edit .CTL Screen

Each line on the screen represents one line in the .CTL file. To change a line, move the cursor over the line, and press F2 or <CR>. The line will then appear in the input area at the top of the screen. Change the line, and then press F2 or <CR> again. This will update the line in PC memory.

Save .CTL File

Figure C-31 illustrates the save .CTL screen.

```
- EDIT - SAVE .CTL
              SAVE .CTL CONTROL FILE ON DISK(ETTE)

     Enter filename:

 Example filenames:

   A:MYFILE   drive=A:

   MYTERM.CTL drive=default

 * filenames may be entered in upper or lower case

 * file extension will default to .CTL

[F2]=Proceed - [F10]=Cancel - Use [Home] [End] to move cursor - [F1]=Help
Alphanumeric Input Mode
```

Figure C-31. Save .CTL Menu

Enter the new name for the .CTL file currently in memory. This will create the .CTL file suitable for linking.

C.11.4 Link

Figure C-32 illustrates the LINK menu.

```
- LINK
                    INITIALIZE FILENAMES FOR 7171LINK

   CONTROL FILE:
   OUTPUT  FILE:

   NOTES:

   CONTROL FILE MUST EXIST - RUN EDIT .CTL TO CREATE ONE

   CONTROL FILE DEFAULT EXTENSION = .CTL

   OUTPUT FILE DEFAULT EXTENSION  = .IMG


[F9] TO BEGIN 7171LINK PROCESSING

[F2] or ↵ to Select for Extract - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-32. 7171LINK Menu

Specify a control file name by selecting the field, and inputting the path and file name of the .CTL file that is to control the linking process. The link will produce a file whose output file name is specified in the second field. That file will be a TERMINAL IMAGE file, and will be ready for I/O to the IBM 7171.

A link report is generated during the process. This report gives the following information:

1. An indication that the .CTL file is in the proper format.
2. An indication of each terminal name in the .CTL file, as it is being linked.
3. Statistical Data comprising the following:
   a. Total number of bytes in the TERMINAL IMAGE file
   b. Total number of bytes saved during the optimization of the terminal tables
   c. Total number of free bytes remaining for more terminal tables.
4. A copy of the VALID TYPES ARE message, as it will appear to terminals connected to the IBM 7171.

C.11.5 I/O

Figure C-33 illustrates the I/O menu.

```
- I/O
                 INITIALIZE PARAMETERS FOR 7171IO

   Function:     _  [ Get | Put ]
   Type:         _  [ Ports | Terms ]
   IMAGE FILE:
   Password:     ________
   Option:       _  [2 if COM2: | leave blank for COM1:]

   NOTES:

   IMAGE FILE DEFAULT EXTENTION = .IMG



[F9] TO BEGIN 7171IO PROCESSING
[F2] or ↵ to Select for Extract - [F10] or [ESC] to Return - [F1] for Help
```

Figure C-33. 7171IO Menu

All information is sent and retrieved from the IBM 7171 via this function. Remember, if the communications port was not initialized during the installation, then I/O between the PC and the IBM 7171 cannot take place. If necessary refer back to the installation procedure. The meaning of the different data fields are as follows:

Function Get or Put. G indicates that this is a transaction to GET information from the IBM 7171, P indicates that this is a transaction to PUT information into the IBM 7171.

Note: Information may only be PUT to the IBM 7171 as long as no terminals other than the one IBM 7171 uses are connected.

Type Ports or Terms. P indicates that the transaction will take place with a PORTS IMAGE FILE, and T indicates that the transaction will take place with a TERMINAL IMAGE FILE.

IMAGE FILE This is the name of the IMAGE file to be transmitted.

Password. This is the ZAP password described in section "Chapter 9. Special Maintenance Facility and System Messages" on page 9-1 of this document. It is initially set to '@@ZAP@@'. It is strongly advised that the user change this password. This password allows special access to the IBM 7171.

Option If transmission is taking place through the COM2: port, then specify '2'.

Note: Before any I/O can take place, the communications port must be initialized through DOS. The settings must coincide with the port configuration set in the IBM 7171 for port 0. The initial setting is for auto baud detect, Even Parity, Seven Data Bits, One Stop Bit. When using dedicated lines, up to 9600 baud may be used. When using a switched network line up to 1200 baud may be used. To transmit at 1200 baud, therefore, the following DOS command must be issued first:

```
MODE COM1:1200,E,7,1
```

The 7171IO program will allow transfer of .IMG files from an IBM PC over communication lines to the IBM 7171. For a dedicated line, an operator only has to invoke the 7171IO program via the menu selection process or by issuing the 7171IO command. If a switched line is used, first an operator must establish the connection over the switched network. The method of dialing into the system is modem dependent. Once the connection is established, then the operator proceeds as for a dedicated line.

As with all commands, this procedure may be implemented in a bat file.

An example of a bat file for automatic dial up is:

```
ECHO .................. >COM1
PAUSE
7171IO    {Get | Put}  {Ports | Term}  filename.IMG password [2]
```

Executing the first command of the bat file outputs the dialing sequence to a modem which has auto dialing capability. The "........" indicates that the user must put in a modem specific character string which will make the auto dial-in connection.

Execution of the PAUSE command will provide the prompt, Strike a key when ready. Press any Key. The Pause command is not required. It is simply a method for delaying execution until the user is ready to continue.

Executing the last command invokes the IBM 7171 Support Utility which inputs or outputs .IMG files into or out of the IBM 7171. If 2 is not specified in the command the input or output will default to PC communications port COM1:.

C.11.6 7171 Messages

The format of the 7171 messages is as follows:

- The utility name appears in capital letters followed by a colon if the message is specific to one utility module otherwise there is no utility name.
- A description of the meaning of the message.

Reminder Messages

Reminder messages inform the user which data types are valid in a given mode of operation.

Alphanumeric Input Mode
This reminder appears upon entering data input mode. The valid alphanumeric input characters are 0-9 and A-Z.

Binary Key Input Mode (see ref manual for details)
7171TERM: This reminder appears upon entering data input mode. Data entry must be the special binary key input described in the manual.

blank name or "ESC" or "F10" will return to menu
7171TERM: This reminder appears after the "Enter Filename:" prompt. These user escape options may be invoked to break processing and to return to the menu.

CORRECT ANY REPORTED ERRORS BEFORE 'PUT' IMAGE TO THE 7171!
7171LINK: This reminder appears in the 7171LINK output listing. Correct any reported errors before using 7171IO to 'Put' the output image to the 7171.

Hex Input Mode: {0 1 ... 8 9 A B C D E F}
7171TERM: This reminder appears upon entering data input mode. The valid hexadecimal input characters are 0-9 and A-F.

... more [Pg Dn]
7171TERM: This reminder message appears in multi-page displays. Press "Pg Dn" to display the next page of a multi-page display.

NOTE: IN-PLACE EDIT, MAX LENGTH = 240 CHARS
7171TERM: This reminder message appears upon entry into the Login Init screen. Data entry is in-place and up to 240 characters may be input into the space provided. Use the cursor movement keys for editing the string.

Numeric Input Mode
7171TERM This reminder appears upon entering data input mode. The valid numeric input characters are 0-9.

Question Input Mode: " Y N " only
This reminder appears when entering data input mode.

Informational Messages

These messages provide the user with information of utility status.

7171 LOGOFF: transmitting Exit cmd
7171IO This informational message appears when 7171IO ends its communications connection to IBM 7171. No user action/response is required.

7171 TERMINAL LIST WILL APPEAR AS FOLLOWS:
VALID TYPES ARE:
7171LINK: This informational message appears in the 7171LINK output listing. If the terminal types do not appear as desired, correct the .CTL input file and relink.

7171MLNK PROGRAM COMPLETED
7171MLNK This informational message appears in the 7171MLNK output listing. No user action/response is required.

7171LINK PROGRAM COMPLETED
7171LINK This informational message appears in the 7171LINK output listing. No user action/response is required.

7171IO PROGRAM COMPLETED
7171IO This informational message appears in the 7171IO output listing. No user action/response is required.

-TOF-
7171TERM: This informational message is displayed while editing Input Strings when the first input string is located at the top of the display. No user action/response is required.

ABEND: transmitting # <illegal> to abort 7171
7171IO This Abnormal end message appears after 7171IO detects a communications or logon error. After the communications link is severed, the user may reconnect and rerun 7171IO.

COM2: will be used for Async i/o"n"
7171IO This informational message appears in the 7171IO output listing if the COM2 optional parameter is input to 7171IO. No user action/response is required.

... Connected
7171IO: This informational message appears in the 7171IO output listing when communications are established to the IBM 7171. No user action/response is required.

Dialup/Connect begun ...
7171IO This informational message appears in the 7171IO output listing when beginning/attempting communications connection to the IBM 7171. No user action/response is required.

TML Source file read begun ...

7171MLNK This informational message appears in the 7171MLNK output listing when TML Source file read begins. No user action/response is required.

Enter 8 character Terminal name:

7171TERM: When this prompt appears, the user must enter the name which is to be assigned to this terminal. Only 8 alphanumeric characters may be used. This name will display in "Valid Types are:" message.

... File closed

7171IO This informational message appears in the 7171IO output listing when the Image file is closed. No user action/response is required.

Get 7171 NV-RAM Size data ...

7171IO GET This informational message appears in the 7171IO output listing when beginning NV-RAM size data receive. No user action/response is required.

Get 7171 Ports data ...

7171IO This informational message appears in the 7171IO output listing when beginning Ports data receive. No user action/response is required.

Get 7171 Term names data ...

7171IO This informational message appears in the 7171IO output listing when beginning Term names data receive. No user action/response is required.

Get 7171 User Tables data ...

7171IO This informational message appears in the 7171IO output listing when beginning User Tables data receive. No user action/response is required.

Get Error: 7171 response not 'Ok'

7171IO GET This informational message appears in the 7171IO output listing when there is a problem communicating with the IBM 7171. Redial and try again.

Image file write begun ...

7171MLNK & 7171LINK: This informational message appears when the output Image file write begins. No user action/response. is required.

Image file write completed

7171MLNK & 7171LINK: This informational message appears when the output Image file is written to disk(ette) and the is closed. No user action/response is required.

LINK PHASE BEGINS ...

7171LINK This informational message appears in the 7171LINKL output listing when link processing begins. No user action/response is required.

LINK PHASE COMPLETED

7171LINK This informational message appears in the 7171LINK output listing when processing is completed. No user action/response is required.

LOAD COMPLETED

This message is displayed when a file has been successfully loaded from disk(ette) into memory. Press any key to continue.

Logoff Exit begun ...

7171IO This informational message appears in the 7171IO output listing when beginning a logoff/disconnect from the IBM 7171. No user action/response is required.

Logoff completed

Dialup: Okay to hang-up now

7171IO This informational message appears when logoff/disconnect is completed. The user may now hang-up a dialed communications line.

no errors found

7171LINK This informational message appears in the 7171LINK output listing when processing has completed. No user action/response is required.

Okay to CLEAR Current data? "Yes or No":

This prompt appears asking if it is alright for this file to replace data currently in memory. Answering Yes to this question will cause the current terminal table information to be overwritten by the new terminal table information being read in from disk. Answering No will allow the user to perform a Save before reinvoking Load.

Put 7171 Ports data ...

7171IO PUT This informational message appears when beginning Ports data transmission to IBM 7171. No user action/response is required.

PASSWORD transmitted

7171IO This informational message appears in the 7171IO output listing when the IBM 7171 password has been transmitted. No user action/response is required.

Put 7171 User Tables ...

7171IO PUT This informational message appears in the 7171IO output listing when beginning User Table data transmission to the IBM 7171. No user action/response is required.

Reading file: xxxxxxxx ...

7171IO PUT This informational message appears in the 7171IO output listing when beginning an Image file read from disk(ette). No user action/response is required.

READING LOAD FILE ...

This message is displayed while a file is being read from disk(ette) into memory. A completion message will be displayed when the operation is completed. No user action/response is required.

REPORT COMPLETED

This message is displayed when a report has been successfully printed. No user action/response is required.

SAVE COMPLETED

This message is displayed when a file has been successfully saved to disk. Press any key to continue.

SAVING FILE ...

This message is displayed while a file is being saved to disk(ette). A completion message will be displayed when the operation is completed. No user action/response is required.

SYNTAX = 7171IO "Get | Put" "Ports | Terms | All | Rom" filename password "2"

7171IO This informational message appears in the 7171IO output listing if 7171IO detects an input parameter syntax error or after an error which MIGHT have been caused by a syntax error. Consult the manual for a detailed description of the required and optional parameters.

Syntax: A> 7171MLNK input.edl output.img >listdev - code = -1

7171MLNK This informational message appears in the 7171MLNK output listing if 7171MLNK detects an input parameter syntax error or after an error which MIGHT have been caused by a syntax error. Consult the manual for a detailed description of the required and optional parameters.

Syntax: A> 7171LINK input.ctl output.img >listdev - code = -1

7171LINK This informational message appears in the 7171LINK output listing if 7171LINK detects an input parameter syntax error or after an error which MIGHT have been caused by a syntax error. Consult the manual for a detailed description of the required and optional parameters.

```
Terminal Tables = nnn bytes
Optimization    = nnn bytes saved
Free space left = nnn bytes
```

7171LINK This informational message appears in the 7171LINK output listing after processing has completed. The statistics may be used to calculate the approximate number of additional Terminal Tables which may be added to the configuration. No user action/response is required.

TYPETERM transmitted

7171IO This informational message appears in the 7171IO output listing when the Terminal name data has been transmitted. No user ASaction/response is required.

Writing file: xxxxxxxx ...

7171IO GET This informational message appears in the 7171IO output listing when beginning an Image file write to disk(ette). No user action/response is required.

'xxxxxxxx' CTRL file scan begun ...

7171LINK This informational message appears in the 7171LINK output listing when control file processing begins. No user action/response is required.

'xxxxxxxx' linked

7171LINK This informational message appears in the 7171LINK output listing for each terminal table linked. No user action/response is required.

'xxxxxxxx' file load begun ...

7171LINK This informational message appears in the 7171LINK output listing as each .TRM file is loaded. No user action/response is required.

===

7171IO triple equal sign is a visual indication that the communications line is clear for transmission. It is seen during 7171IO connect and disconnect transmission sequences. No user action/response is required.

Warning Messages

CROSS-CHECK: FEAT (Delay) = 0 ms

7171TERM 7171LINK This warning message appears when a delay was specified in one or more output strings for a terminal and the value specified for delay in the terminal features is zero. Input a delay value in the Features screen.

INCOMPLETE DATA SAVED

This warning message appears when a file save was not completed due to a DOS file write error. Be aware that the output file contains only a part of the information which was to be saved. No user action/response is required (other than caution).

INCOMPLETE OR ERRONEOUS DATA LOADED

This warning message appears when a file read was not completed due to a DOS file write error. Be aware that the loaded data contains only a part of the information which was to be loaded. No user action/response is required (other than caution, particularly in saving the incomplete data back to the same file).

INPUT ERRORS FOUND IN LOAD DATA

7171TERM This warning message appears when input errors are found in load data. Either the file is not formatted as a .TRM file or it contains illegal data. Note that the load has completed, but some string data may be missing. These errors will only occur if the file is either not a .TRM file (wrong file loaded) or if a .TRM file has been edited by a non-7171 program.

RESET CHAR(S) FOUND

7171TERM This warning message appears when one or more reset characters is used in an input string. This will result in the string being unusable because reset characters have priority over input strings. The user should either eliminate the reset character(s) from the input string or redefine the reset character(s).

THIS RESET CHAR FOUND IN ONE OR MORE INPUT SEQ STRINGS

7171TERM This warning message appears when a character specified as a reset character was used in one or more input strings. This will result in those strings being unusable because reset characters have priority over input strings. Eliminate the reset character from the input string(s) or redefine the reset character.

WARNING: Cursor Origin > 127

7171TERM This warning message appears when the cursor origin in Terminal Features exceeds the maximum value allowed. Valid origins must be between 0 and 127. Reinput a valid cursor origin.

WARNING: DELAY > 255 ms

7171TERM This warning message appears when the delay in Terminal Features is greater than 255 msecs. Normally delay values are between 0 and 255 msecs. The value specified is probably byte reversed. The user should either redefine the delay value or ignore this warning (assuming the value IS correct).

Error Messages

nn FATAL ERRORS ENCOUNTERED

7171LINK This summary error message appears in the output listing. Correct all fatal errors before PUT Image to IBM 7171.

nn FEATURE ERROR(S) FOUND

7171LINK This error message appears in the output listing. Individual error/warning messages are shown for each error found by 7171LINK.

nn INPUT SEQ ERROR(S) FOUND

7171LINK This error message appears in the output listing. Individual error/warning messages are shown for each error found.

nn LOGIN SEQ ERROR(S) FOUND

7171LINK This error message appears in the output listing. Individual error/warning messages are shown for each error found.

nn OUTPUT SEQ ERROR(S) FOUND

7171LINK This error message appears in the output listing. Individual error/warning messages are shown for each error found.

nn RESET CHAR ERROR(S) FOUND

7171LINK This error message appears in the output listing. Individual error/warning messages are shown for each error found.

1st CHAR NOT ASCII CTRL CHAR

7171TERM The first character of all usable input strings must begin with a valid ASCII control character (00-1F). The first character of the specified input string is not valid. Redefine the input string.

7171MLNK: error opening 'xxxxxxxx' input file

7171MLNK This error message appears in the output listing when the named input file is not found or, rarely, if there is a disk(ette) i/o error. Reinvoke 7171MLNK with the input file parameter naming an existing file and be sure the diskette drive and (sub)directory are correct.

7171MLNK: error opening 'xxxxxxxx' output image file

7171MLNK This error message appears in the output listing when a disk(ette) directory is full,does not exist, or, rarely, if there is a disk(ette) i/o error. Reinvoke 7171MLNK with the output file parameter naming an another file and be sure the diskette drive and (sub)directory are correct. If writing to a diskette, check it for a write-protect tab.

7171MLNK ABEND: *> DOS FILE ERROR WRITING IMAGE FILE * - code = nn

7171MLNK This abnormal end error message appears when 7171MLNK terminates due to a DOS file error. If writing to a diskette, check it for a write-protect tab.

7171IO: Error Opening 'xxxxxxxx' file

7171IO This error message appears in the output listing for both read and write files. Reinvoke 7171IO with the file name spelled correctly and be sure the diskette drive and (sub)directory are correct. If writing to a diskette, check it for a write-protect tab.

7171IO ABEND CODE = nn

This abnormal end error message appears in the output listing at the end of 7171IO processing. (See the 7171IO Codes table).

7171IO utility apparently out-of-sync with IBM 7171 *
Buffer= (buffer data)
Abort, Retry, or Ignore?:**

7171IO This prompt appears when the 7171IO utility is apparently out-of-sync in communicating with the IBM 7171. Buffer data is for diagnostic information only. Enter "R" for Retry or "A" for Abort and redial.

7171LINK ABEND: CONTROL FILE SCAN ERROR - code = nn

7171LINK This abnormal end error message appears in the output listing when the control file does not have the correct format and/or the control file contains invalid data. Consult the manual for the correct format and re-edit the .CTL file before relinking.

7171LINK ABEND: *> DOS FILE ERROR WRITING IMAGE FILE * - code = nn

7171LINK This abnormal end error message appears in the output listing when 7171LINK terminates due to a DOS file error. If writing to a diskette, check it for a write-protect tab.

7171LINK ABEND: LINK PHASE ERRORS - code = nn

7171LINK This summary error message appears in the output listing. The user must correct all errors before PUT Image to IBM 7171.

7171LINK: error opening 'xxxxxxxx' input file

7171LINK This error message appears when the named input file is not found or, rarely, if there is a disk(ette) i/o error. Reinvoke 7171LINK with the input file parameter naming an existing file and be sure the diskette drive and (sub)directory are correct.

7171LINK: error opening 'xxxxxxxx' output image file

7171LINK This error message appears in the output listing when a disk(ette) directory is full, does not exist, or, rarely, if there is a disk(ette) i/o error. Reinvoke 7171LINK with the output file parameter naming an another file and be sure the diskette drive and (sub)directory are correct. If writing to a diskette, check it for a write-protect tab.

-EOF- ALL STRING SEQ USED: NO MORE ROOM FOR APPENDS

7171TERM This error message appears when the maximum allowable number of input strings has been entered. There is no room to add more input strings without deleting previous strings.

 # init XMT ERR=nnnn + xxxx

7171IO (abend -1) This error message appears when there are communications problems. Hexadecimal status is BIOS INT 14h status for diagnostic information only. Redial and try again.

ABEND DURING ECHOBACK: (buffer data)

7171IO (abend -5) This abnormal end error message appears when there are communications problems. Buffer data is for diagnostic information only. Redial and try again.

ABEND DURING IORCV: (buffer data)

7171IO (abend -6) This abnormal end error message appears when there are communications problems. Buffer data is for diagnostic information only. Redial and try again.

ABEND DURING IOXMT: (buffer data)

7171IO (abend -5) This abnormal end error message appears when there are communications problems. Buffer data is for diagnostic information only. Redial and try again.

ASYNC BREAK ERROR ***

7171IO This error message appears when a Break is detected. The communications line is probably down or subject to intermittent breaks in satellite transmissions.

ASYNC FRAME ERROR ***

7171IO This error message appears when there is a Framing error detected. The communications line is probably down or subject to problems in satellite transmissions.

ASYNC PARITY ERROR ***

7171IO This error message appears when there is a Parity error detected. The parity error may be due to a noisy communications line or loose cable connection. Check cables and redial to try to get a cleaner communication line.

AUTOBAUD XMT ERR=nnnn + xxxx

7171IO (abend -2) This error message appears when there are communications problems. Hexadecimal status is BIOS INT 14h status for diagnostic information only. Redial and try again.

NV-RAM OVERFLOW - TOO MANY TERMINALS

7171LINK This error message appears in the output listing when 7171LINK determines that the output Image would overflow NV-RAM. Relink using fewer terminals in .CTL file or decrease the number and/or length of strings in one or more terminal tables.

CROSS-CHECK ERROR: FEAT (Flags) vs OUTPUT (Display Mode) vs GRAPHIC

7171TERM This error message appears when one or more of the following is not specified: Feature Flags highlighting, Display Mode Output Sequence, or Graphic Rendition Control String. Either specify all three or disable all three disabled (no highlighting).

DOS FILE ERROR ENCOUNTERED

This error message is displayed when an error is encountered while reading or writing to a disk(ette). The operation did not complete successfully. If writing to a diskette, check it for a write-protect tab.

DOS FILE ERROR ENCOUNTERED: INCOMPLETE REPORT

This error message appears when there is a printer error or a file save was not completed due to a DOS file write error. If writing to a diskette, check it for a write-protect tab.

ERROR OPENING ...

This error message appears when an error was encountered attempting to open a specified file. This may be because the file does not exist on the specified drive. Reinvoke with the file name spelled correctly and be sure the diskette drive and (sub)directory are correct. If writing to a diskette, check it for a write-protect tab.

ERROR: ONE OR MORE GRAPHIC CHARS NOT 7-BIT ASCII

7171TERM This error message appears when one or more Graphic Rendition Control characters are invalid. All Graphic Rendition Control characters must be valid ASCII characters (00-7F). Redefine the invalid character(s).

FATAL ERROR LOADING: GRAPHIC RENDITION CHARS

7171LINK This error message appears when IBM 7171LINK detects an error during a file load. Relink after editing Graphic Renditiön Control String in 7171TERM.

FATAL ERROR LOADING: INPUT STRINGS

7171LINK This error message appears when 7171LINK detects an error during a file load. Relink after editing Input Strings in 7171TERM.

FATAL ERROR LOADING: LOGIN STRING

7171LINK This error message appears when 7171LINK detects an error during a file load. Relink after editing Login Init Sequence in 7171TERM.

FATAL ERROR LOADING: MORE DATA THAN EXPECTED

7171LINK This error message appears when 7171LINK detects an error during a file load. The .TRM file format is bad. Relink after saving from 7171TERM.

FATAL ERROR LOADING: OUTPUT STRINGS

7171LINK This error message appears when 7171LINK detects an error during a file load. Relink after editing Output Sequences in 7171TERM.

FATAL ERROR LOADING: RESET CHARS

7171LINK This error message appears when 7171LINK detects an error during a file load. Relink after editing Reset Characters in 7171TERM.

FATAL ERROR LOADING: TCD DATA

7171LINK This error message appears when 7171LINK detects an error during a file load. Relink after editing Terminal Features in 7171TERM.

FATAL ERROR LOADING: TERMINAL NAME

7171LINK This error message appears when 7171LINK detects an error during a file load. The .TRM file format is bad. Relink after saving from IBM 7171TERM.

FILE READ or FILE FORMAT ERROR

7171IO PUT This error message appears when 7171IO detects an error reading Image file. Relink the file and correct any reported link errors.

FILE WRITE ERROR

7171IO GET This error message appears when 7171IO detects an error writing to an Image file. Retry the 7171IO and write to another disk(ette).

ILLEGAL CHAR(S) FOUND

7171TERM This error message appears when one or more characters are greater than 80 hex and are not valid special 8-bit control characters (bin__x, char__x, etc.). Legal characters are 7-bit ASCII (00-7F) and the special 8-bit control characters.

Insert error - Press "Ins" key

This error message is displayed when an input area is full and characters are being entered while in insert mode. Press the "INS" key to exit insert mode.

INVALID BAUDRATE

7171TERM This error message appears when a baud rate is entered which is not one of the allowable values. The valid baud rates are listed on the Configure 7171 Ports screen.

INVALID TERMTYPE: xxxxxxxx

IBM 7171TERM This error message appears when the terminal name xxxxxxxx is not found in the TERMTYPS list in the Image file which was loaded. Refer to the Link output listing for the list of valid terminal types.

IOXMIT TIMEOUT ERR; STATUS = xxxx

7171IO (abend -5) This abnormal end error message appears when there are communications problems. Hexadecimal status is BIOS INT 14h status for diagnostic information only. Redial and try again.

LOGON ABEND: 'Error' expected; Buffer = (buffer data)

7171IO (abend -3) This abnormal end error message appears when there are communications problems. Buffer data is for diagnostic information only. Redial and try again.

NO ASYNC DATA FOR 5 SECS

7171IO (abend 5) This abnormal end error message appears when there is no data received from IBM 7171 for 5 seconds (line is still up). This message indicates a possible IBM 7171 problem.

NO Data Set Ready FOR 5 SECS

7171IO This error message appears when the modem is not turned on or if the user forgot to dialup. Correct the problem and redial.

NO TERMINAL NAMES IN IMAGE

7171PORT This error message appears when a file is either not an Image file (wrong file loaded) or is the output of a bad link. If this is output from a TML Link, there is no TERMTYPS statement in the TML source.

NO TERMTYPS FOUND - IS YOUR CONTROL FILE EMPTY ? *

7171LINK This error message (not a prompt) appears when there are no terminal tables specified in the input control file. Re-edit the control file and then relink.

ONE OR MORE PARAMETERS NOT SPECIFIED

7171 & 7171TML This error message appears when there is an error loading the 7171IO or 7171MLNK subutilities. The user did not specify one or more required parameters. Return to the screen and input ALL required parameters.

OVERRUN ERROR ***

7171IO This communications error message appears when Async data is being received faster than the PC can process it. Try again using a lower baud rate.

RESET CHAR MUST BE AN ASCII CTRL CHAR

7171TERM This communications error message appears when the value specified is not a valid Reset character. All Reset characters must be ASCII control characters (00-1F). Redefine the invalid Reset character.

RCV ERR=nnnn Status= xxxx

7171IO (abend -6) This error message appears when there are communications problems. Each x corresponds to a hexadecimal digit such that for ERR, the first byte should equal the Line Control Status:

- 01 = Data Ready
- 02 = Over run Error
- 04 = Parity Error
- 08 = Framing Error
- 10 = Break Detected
- 20 = Transmit Holding Register empty
- 40 = Transmit Shift Register empty
- 80 = Time Out

The second byte of ERR equals the character received. The first byte of STATUS equals the Line Control Status, and the second byte equals the Modem Status:

- 01 = Delta Clear to Send
- 02 = Delta data set ready
- 04 = Trailing edge ring detector
- 08 = Delta receive line signal detect
- 10 = Clear to send
- 20 = Data set ready
- 40 = Ring indicator
- 80 = Received line signal detect

TOO MANY TERMTYPS, MAX = nn *

7171LINK This error message appears in the output listing when the 7171LINK TERMTYPS table overflows. Relink using fewer terminals in the input .CTL file.

UNEXPECTED IBM 7171 RESPONSE: (buffer data) - Expected 'Ok' or 'Error'

7171IO This error message appears when there is a problem communicating with the IBM 7171. Buffer data is for diagnostic information only. Try again.

XMIT = xxxx & ECHO = xxxx

7171IO (abend -5) This error message appears when there are communications problems. Each x corresponds to a hexadecimal digit such that for XMIT, the first byte should equal 00, and the second byte should equal the character that was sent. For ECHO, the first byte equals the Line Control Status:

- 01 = Data Ready
- 02 = Over run Error
- 04 = Parity Error
- 08 = Framing Error
- 10 = Break Detected
- 20 = Transmit Holding Register empty
- 40 = Transmit Shift Register empty
- 80 = Time Out

The second byte of ECHO equals the character received.

xxxxxxxx DUPLICATE TERMINAL NAME LINKED - non-fatal error, processing continues

7171LINK This error message appears in the output listing when a terminal table has the same terminal name as a previously linked table. The output image MAY be partially usable, but only the first terminal table will be accessible (the duplicate(s) will not be accessible).

7171IO Codes

The following section explains the meaning of error codes which are shown during the operation of the IBM 7171 Support Utility.

```
7171IO Abend codes

   -1 initial '#' transmission
   -2 initial [CR] for auto baud detect
   -3 initial '?' for verification (after password & TYPETERM)

   -4 during transmission
   -5 during echoback
   -6 during receive

   -12 unexpected 7171 Dump response (7171 problem)
   -13 unexpected 7171 Dump data     (7171 problem)
   -99 unexpected 7171 Dump response (7171 problem)
```

C.12 The IBM 7171 Macro Language

IBM 7171 Support provides another method of generating a terminal .IMG file which can be loaded into the IBM 7171. This method of generating a terminal .IMG file uses the IBM 7171 Macro Language. First, the Macro Language file is created using an Editor such as EDLIN or the IBM Personal Editor. Use an extension of .TML for the Macro Language files. Specifically allowable IBM 7171 Macro Language statements are those found in the IBM Manual SB30-1911 *IBM Series/1 Yale ASCII Terminal Communication System II Program Description/Operations Manual* (including TNL SN60-1215 of 9/15/83) with the limitations discussed below. Once all the terminal tables are defined they are linked together into a IBM 7171 loadable .IMG file using the MLNK function. The following macro statements are supported:

TERMTYPS

EADSTERM ORIGIN= HTRAN= TTRAN= DELAY= RCHRS= SGR= FLAGS=

EADSCSS EQU=

EADSNODE CALL GOTO PARM=

label EQU * & another label (including ROM vector names)

TITLE SPACE SKIP & * comment (documentation)

END (logical e-o-f for assembler source)

Error messages will be generated by MLNK for the following parms:

PAD=

PC TML Syntax Templates

This describes the general syntax for PC TML statements.

PC TML statements may be:

- 1 or more lines in length
- Line lengths are up to 80 bytes
- Statement lengths may be up to 350 bytes
- Statement extension is possible through the keyword CONTINUE

```
label        macro        parms         comments

* comment line
```

- LABEL

  - Depending on the macro, label may be required or optional.
  - Labels always begin in column 1.
  - Labels are terminated by a space or tab.
  - Maximum Label length is up to 13 characters.
  - Label names are converted to uppercase during processing.

- MACRO

  - Identifies the macro which will be used.
  - The Macro name is separated from the label by a space, tab.
  - The maximum length of a macro name is 8 characters.
  - The macro name is converted to uppercase for processing

- PARMS

  - Depending on the macro, this field may not be required.
  - Parms are separated from a Macro name by a space, tab, or a comma.
  - The parm list is terminated by a space, tab, or eol. [eol] is a constant denoting the end of a line, CR, LF.
  - The maximum length of a parm list is 52.
  - Individual parms are separated by a comma.
  - Quoted strings may be parms if delimited by single quotes, ' ' and may contain lowercase characters which will be processed as lowercase.
  - Unquoted parms are converted to uppercase during processing.

- COMMENT

  - This field is optional and is separated from PARMS by a space, or a tab.
  - Comment fields are terminated by the eol.
  - The comment field will prevent line continuation.
  - The comment field is ignored during processing.

- COMMENT LINES begin with a STAR in column 1 and may appear anywhere.

C.12.1 Macro Syntax

Each of the macro statements in the following sections has its syntax described according to the following conventions:

- Words in capital letters are keywords and must be entered as shown. They may be entered in any combination of uppercase and lowercase. PC TML converts words to uppercase, except when they are part of a quoted string.

- Items in square brackets, [ ], are optional.

- An ellipsis ...... indicates that an item may be repeated, provided rules with respect to length and syntax are observed.

- The | denotes a choice between the item before and after it.

- Uppercase letters and punctuation marks other then those described above, represent information which must appear exactly as shown.

The PC TML macro specifications which follow are flagged with differences from the EDL implementation.

```
Key:       -  Same as EDL
           +  Enhancements to EDL
           *  Different meaning than EDL
```

```
[label]    TERMTYPS    name1,name2,....[+|-]
```

```
The meaning of each field is described below:

      label      + optional dummy label
                 + ignored if present, useful for documentation only

      TERMTYPS   - The Macro's name

      names      - must be labels of TERM macros or symbols equated to
                   labels of TERM macros
                 * Equates for ROM terminal names are predefined,
                   these names cannot be overridden

      Several consecutive TERMTYPS macros may be coded to provide
      control over the display of the ENTER TERMINAL TYPES message.
      All but the last TERMTYPS statement may end with a +
      or - character.
      This macro and the extended form are identical to the EDL
      implementation.
```

TERM Macro

```
                                                 Defaults:
[label]     TERM         inputtab
                       * [,ORIGIN=ochar]          X20
                         [,HTRAN=hosttable]       + HTRAN (in ROM)
                         [,TTRAN=termtable]       + TTRAN (in ROM)
                         [,DELAY=nnn]             - 0
                       * [,RCHRS=stringname]      + RCHS (in ROM)
                       * [,SGR=stringname]        + SGR (in ROM)
                       * [,FLAGS=X....char]     - X0000
```

```
The meaning of each field is described below:

    label     - required to assign a symbolic name for TERMTYPS
              + only the first 8 characters will be used for TERMTYPS

    TERM      - The Macros name

    inputtab  - Label of the NODE macro which define the start of
                the first level input parse table
              + May be a ROM equate name

    ORIGIN    - Keyword which provides a base to be used in generating
                direct cursor addressing strings
              - Default is X20
              + Cx equals any valid ASCII character

    HTRAN     - Keyword permits different terminal types to be
                assigned to different character translate tables
              + May be a ROM equate name (default)

    TTRAN     - Keyword permits different terminal types to be
                assigned to different character translate tables
              + May be a ROM equate name (default)

    DELAY     - Keyword specifies in milliseconds, the length
                of the pauses to be inserted into output strings as
                specified in the CSS macros for this terminal type

    RCHRS     - Keyword points to a string of special reset keyboard
                characters which provide the user control over error
                recovery, and define pacing protocol.  The definition
                of the RCHRS string is described in the CSS section.
              * Note that PC RCHRS are defined differently than
                EDL's  RCHRS

    SGR       - Keyword points to a string of special graphic rendition
                control characters which provide the user control over
                intensity and color of  the 3270 field attributes

    FLAGS     - Keyword provides bit flags which can be set to
                describe special terminal characteristics
              - Flag definitions are :
                X8000 Hardcopy
                X4000 APL / Non-APL terminal (SI/SO)
                X1000 Bit paired APL terminal
                X0800 Highlighting can be performed
                X0400 Highlighting is a mode
                X0040 Character display in lower right corner
                             without scroll
                X0020 Uses 3270 algorithm for nulls on Read Modified
```

CSS Macro

```
[label]    CSS   char1,char2,....

[label]    CSS   EQU=label2

[label]    CSS   char1,char2,....[,CONTINUE]
[label]    CSS   [CONTINUE,]char1,char2,...

[label]    CSS   char1,char2,....[,XX]
[label]    CSS   [XX,]char1,char2,...
```

```
Notice there are several forms of the CSS Macro.
The meaning of each field is described below:

     label      + Up to 13 characters in length

     CSS        - The Macro's name, an alias of EADSCSS

     chars      - May be ASCII equate names(i.e. ESC, LBRACK,..)
                + May be a Quoted string(i.e. 'LOGON')
                + May be a Character string(i.e. C'LOGON')
                + May be a Hexadecimal string(i.e. X'0123')
                * See section on converting EDL to TML files for
                  undefined ASCII character equates

     CONTINUE   - Keyword which continues the CSS definitions
                - Last name on one CSS statement and the first name
                  on the next

     XX         + A synonym for CONTINUE, the usage is the same

     EQU        - Keyword to equate the value of 2 CSS labels
```

C.12.2 IBM 7171 Support Utility Macro Language Versus Series/1 IUP

Here is a table which compares the Series/1 Macro language to the IBM 7171 Macro Language. The table is meant to present differences in the two languages as well as similarities. A '-' denotes that a specific parameter or statement does not have an equivalent. References to the PORTS menu, are there to show where unsupported statements can be emulated.

<table>
<tr><th>Series/1</th><th>7171TML</th><th>Comments</th></tr>
<tr><td>TERMINAL DEVICE=EADS,ADDR=xx
BITRATE=linspeed
ATTN=option
LINSIZE=printwidth
CODTYPE=ASCII
PART=n</td><td>-
-
-
-
-
-</td><td><s/1 specific>
PORTS menu
<s/1 specific>
<s/1 specific>
<s/1 specific>
<s/1 specific></td></tr>
<tr><td>EADSLINE PAGESIZ=rows
LINESIZ=cols
TYPE=termtype
XPNT=yes
LAST=yes</td><td>-
-
-
-
-</td><td>PORTS menu
PORTS menu
PORTS menu
not implemented
<s/1 specific></td></tr>
<tr><td>TERMTYPS ...</td><td>TERMTYPS ...</td><td></td></tr>
<tr><td>EADSTERM inputtab
outputtab
ORIGIN=ochar
PAD=(p1,p2,p3,p4)
HTRAN=hosttable
TTRAN=termtable
DELAY=nnn
RCHRS=stringname
SGR=stringname
FLAGS=X'....'</td><td>TERM inputtab
outputtab
ORIGIN=ochar
-
HTRAN=hosttable
TTRAN=termtable
DELAY=nnn
RCHRS=stringname
SGR=stringname
FLAGS=X'....'</td><td>

<s/1 specific></td></tr>
<tr><td>EADSCSS parm1,parm2,...
EQU=
(CHARY)(CHARX)(BINY)(BINX)
(HARDCOPY)(DELAY)
(CHARFLD)(BINFLD)
(USERY)(USERX)</td><td>CSS parm1,parm2,...
EQU=
(CHARY)(CHARX)(BINY)(BINX)
(HARDCOPY)(DELAY)
(CHARFLD)(BINFLD)
(USERY)(USERX)</td><td></td></tr>
<tr><td>EADSNODE char,[CALL | GOTO] ...</td><td>NODE char,[CALL | GOTO] ...</td><td></td></tr>
</table>

C.12.3 TML Syntax Restrictions

SYNTAX RESTRICTIONS	EDL	TML	TML Error Code
Maximum line length in bytes	72	80	# 101
Maximum Statement length in bytes	350	350	# 103
Maximum Parms field length in bytes	350	64	# 208
Maximum Label name length in characters	8	13	# 104
Maximum macro name length	8	8	trunc
Maximum number of labels	16,383	256	# 200
Maximum number of label references	N/A	512	# 202
Continuation Column	72	any column	

Figure C-34. A Comparison of EDL and TML

C.12.4 TML Pseudo Op Comparison

PSEUDO-OP	EDL ASSEMBLER	TML EMULATION
DC	8 characters(max)	64 characters(max)
DATA	yes	yes-enhanced
EQU	yes	yes
END	yes	yes
EJECT	yes	no
SKIP	yes	no
TITLE	yes	no

C.12.5 Differences and Restrictions

1. MLNK does not support TERMINAL and EADSLINE functions. The IBM 7171 supported functions are inputted and maintained interactively under the PORTS menu.

2. TERM parameters:

   Inputtab labels may reference IBM 7171 equate names.

   PAD=... does not generate data, but will generate a warning message.

   HTRAN= and TTRAN= default to IBM 7171 ROM translate tables. To support customized tables, the entire table(s) must be defined in the .TML file. Both tables combined will require just over 2K of NV-RAM and may limit the number of user-defined termtyps which will fit in the remaining NV-RAM space.

DELAY= has the same meaning for the IBM 7171 as for the Series/1.

ORIGIN= provides the same capabilities plus the C'x' form.

The IBM 7171 RCHRS string has 1 less character (8 vs 9) and the pacing character positions are not the same. EDX attention and "PA4" are Series/1 specific. Also, there is no default RCHRS stringname "EADSRCHRS" - the user-defined name must be explicitly coded.

The IBM 7171 SGR string is identically defined as for the Series/1 except that there is no default SGR stringname "EADSSGR" - the user-defined name must be explicitly coded.

FLAGS= bits have the same meanings as for the Series/1 except for the flag X'0080' for NULLS which is Series/1 specific.

3. CSS strings are limited to 16 bytes as for the Series/1 (PDOM page 3.26.3).

4. CSS 1st string (Reposition) parameters (USERX) and (USERY) are are supported.

5. CSS 18th string (Display Mode) uses (CHARFLD) (BINFLD) in the same manner as for the Series/1.

6. MLNK differences from $EDXASM:

   TITLE, SKIP, EJECT commands are allowed but are ignored (no-op).

   END is required.

   DC and DATA statements have been enhanced.

   The only other (than above) statements supported are: TERMTYPS, TERM, CSS, and NODE.

   LINE CONTINUATION: if the parm field ends with COMMA, EOL. Note that $EDXASM convention of continuation character in col 72 will be treated as a comment and not as a continuation character (see below).

   COMMENT FIELD: must follow a parm field, and prevents continuation if not on last line.

   COMMENT LINES: any line beginning with a STAR, '*' is ignored, even within a multi-line statement.

   FIELD DELIMITERS: commas are acceptable between parms, between command and parm, and between label and command.

   PARMS: are delimited by commas and ended by white space or EOL; single quotes enclose strings which may contain commas and white space.

CAPITALIZATION: except for alpha characters in quoted strings, all lower case alpha is converted to upper case.

CSS parms may be asciiname, hexstring, decimal number, or CSS (equate) as for Series/1.

C.12.6 EDL to TML Conversion

Conversion of EDL files to TML files can be accomplished by doing the following:

1. Eliminate line numbers in col 73-80

2. Search for continuation chars in col 72

   - Eliminate continuation char
   - Eliminate or move any comment between parms and col 72. If the user chooses to move a comment, there are two places the comment may be placed:
     - On a separate line with a STAR, '*', in column 1.
     - In th comment field of the last line of a statement
   - The continuation line may begin in any column by simply placing a COMMA, ',', after the last parm on the line.

   Here is an example of an EDL file before the comment change, and after the comment change(2 variations):

Column #	1 9 16 72 80

EDL	`label   macro  op1,    * comment1          $00000001`
	`               op2,    * comment2          $00000002`
	`               op3     * comment3          $00000003`

TML	`label   macro  op1,`
	`               op2,`
	`               op3     * comment1 comment2 comment3`

TML	`label   macro  op1,`
	`*                         comment1`
	`               op2,`
	`*                         comment2`
	`               op3      * comment3`

3. All terminal names which equate to ROM terminal names must be changed if they are to be redefined in PC TML. Here are the ROM terminal names and their values, see also Appendix F for a ROM listing.

7171 ROM EQUATES							
TERMTYPS		INPUT TABLES		SPECIAL		GENERAL	
Addr	Equate	Addr	Equate	Addr	Equate	Addr	Equate
8000	IBM3101	8035	IBM3101I			8025	BITMATCH
8002	TVI912	805F	TVIDI			802F	COMSET
8003	TVI920	805F	TVIDI			8029	FUNBPFK
8004	TVI950	805F	TVIDI			802B	FUNCA
8005	TVI950R	8063	TVII			802A	FUNCX
8006	ADM31	8058	ADM31I	8032	ADM31ATR	8028	FUNCPFK
8007	ADM3A	8055	ADM3AI	8033	ADM31MOD	8071	HTRAN
8008	VT100	8047	VT100I	8053	VTPFKS	8022	QWERTY1
8009	DM1520	803C	DM1520I			8023	QWERTY2
800A	DM1521	803C	DM1520I			8024	QWERTY3
800B	DM3045	8040	DM3045I	8031	DM3SGR	8021	RCHRS
800C	TYPETERM	806A	TPTMI			802C	RSETCHAR
800D	HARDCOPY	806D	HCPYI			802D	SETCHAR
800E	PLOTTER	806D	HCPYI			802E	SETMTCH
						8030	SGR
						8026	TYPPFK
						8070	TTRAN

Figure C-35. ROM Equates

4. Change EADSTERM statements to TERM
   - Outputtab positional parm not supported in TML
   - ORIGIN= default is X'20'
   - HTRAN=,TTRAN=,SGR=,RCHRS= default to ROM equates
   - FLAGS=X'0080' & X'0010' not supported on 7171
   - PAD= not supported on 7171, will generate a warning
5. Change the EADSCSS statements to CSS
6. Change the EADSNODE statements to NODE
7. Change the RCHRS statements to the 7171 definition
8. Change the ASCII mnemonics as directed here:
   - Change TAB to HT
   - Change LINEFEED to LF
   - Change FORMFEED to FF
   - Change DC1 to XON
   - Change DC3 to XOFF
   - Change ESCAPE to ESC
   - Change BLANK to SPACE
   - Change ASTERISK TO STAR
   - Change HYPHEN to MINUS
   - Change DOT to PERIOD

C.12.7 MLNK Error Code Conventions

Fatal errors terminate processing.

Non-fatal errors may terminate processing of the current line or statement, but processing will continue with the next line or statement.

Negative codes are fatal errors.

Positive codes are non-fatal errors.

-100 = Unexpected end-of-file	= END statement not coded
101 = Input line truncated	
102 = Label on continuation line	
103 = Statement too long	> 350 bytes
104 = Too many fields in statement	> 52 operands
105 = Unmatched quotes	
106 = Required fields missing	
200 = Label table overflow	> 256 label definitions
201 = Duplicate label	Remember, ROM equate names are labels, if the user uses the same name, they become duplicate labels. See Figure C-35 on page C-77.
202 = Reference table overflow	> 512 label references
203 = Unresolved reference(s)	= undefined label(s)
204 = Unrecognized command	YALE MACRO statement NOT supported
205 = Core buffer full	> 8K NV-RAM exceeded
206 = Invalid operand	
207 = Unmatched quote	
208 = Operand too long	> 64 bytes
209 = Invalid digit	
210 = Odd number of digits in hex string	
211 = Operand missing	

212 = Label required

213 = Invalid label reference

214 = Last terminal missing CSS string(s)

215 = 'EQU=' parm on continued CSS statement

= Unrecognized error (shouldn't happen)

Appendix D. Hints on Using ASCII Printers and Plotters

Printers and plotters attached to the IBM 7171 have successfully been controlled using the transparent mode interface with device line drivers running on the host under VM and RSCS.

Assistance in setting up ASCII printers and plotters and information on line drivers will be supplied in a later release of this document.

Appendix E. Interface for User Supplied Table Modification Program

E.1 Interface Description

An interface exists for a user supplied table generation program. The interface provides defined input and output data streams. The interface may be written for any computing device that provides the standard RS-232-C interface that is described in "Chapter 5. IBM 7171 I/O Interface to Terminals" of this document. The user may write his own program to utilize this interface. However, the user will be responsible for all consequences of using the interface. This program will be referred to as the 'User Program' throughout this Appendix.

The interface is intended to be used to modify, or add Terminal Definition Tables that are not supplied with the IBM 7171, or to change the default port configuration. The following commands are supplied:

Dump	Retrieve data from the IBM 7171
Compressed Dump	Retrieve compressed data from the IBM 7171
Store	Store data to the IBM 7171
Exit	Exit the IBM interface program

E.2 The IBM 7171 Interface Initialization

The User Program must first initialize the communications line as described in "Chapter 5. IBM 7171 I/O Interface to Terminals." The terminal that runs the User Program must be connected to TC 0 port 0, with no other terminals being logged on. The IBM 7171 will then present the 'ENTER TERMINAL TYPE:' message. The User Program must then present the password '@@ZAP@@' to the communications line, followed by a Carriage-Return. The IBM 7171 will then prompt again with 'ENTER TERMINAL TYPE:', to which the User Program must respond with the terminal type 'TYPETERM', with no Carriage Return. The IBM 7171 will then respond with the initialization sequence: Carriage Return, Form Feed (X'0D0C').

All values for storage bytes are byte-reversed as describe in "Chapter 4. Customizing IBM 7171 Tables." They are dumped and stored exactly as they appear in memory. All values for segments and offsets are not byte reversed. The expression '1000' used as an offset is interpreted as 4096 decimal.

E.3 Definition of the IBM 7171/User Program Interface

In the following definitions, this notation applies:

addr An ASCII representation of a 2 byte address, e.g. 1C00. The digits 0-9 (ASCII X'30' - X'39') and the letters A-F (ASCII X'41' - X'46') are legal. All addresses are offset from segment DC00.

CRLF The ASCII Carriage Return - Line Feed sequence, i.e. X'0D0A'

CR The ASCII Carriage Return, i.e. X'0D'

bytebyte The ASCII representation of any two arbitrary bytes, e.g. 1F43. The digits 0-9 (ASCII X'30' - X'39') and the letters A-F (ASCII X'41' - X'46') are legal. In the special case where the 'bytebyte' is the LAST of a 'one or more'(!) sequence, one or two bytes may be used. e.g. "S 1C00: 3456 7890" and "S 1C00: 3456 1432 0A3B 78" are both legal but "S 1C00: 3456 12 4325" is NOT legal.

chksum A one byte checksum of all data bytes on the line just issued in response to dump commands. One line is defined as the data bytes between the second colon and the checksum in a DUMP response. The checksum algorithm is given later.

! One or more of the sequence immediately following the !. The sequence may or may not be surrounded by parentheses.

' Characters between single quotes are literal ASCII characters.

E.4 Interface Commands

The commands are defined as follows:

1. Exit

   a. Input

      To issue an exit command, enter:

      ```
      ('E') + (CR)
      ```

   b. Output

      The response is

      ```
      (CR) + ('Ok') + (CR)
      ```

      Followed by the 'ENTER TERMINAL TYPE:' message.

2. Store

   a. Input

To issue a store command, enter:

```
('S') + (' ') + (addr) + (':') + !(' ' + bytebyte) + (CR)
```

This takes the bytes represented by the 'bytebytes' and stores them sequentially in NV-RAM starting at the given addr.

b. Output

(CRLF) + ('Ok') + (CRLF)

c. Notes:

1) No input line may exceed 80 characters, including the command and the data.

2) The ZAP program will echo back the entered characters for the store command. It is the User Program's responsibility to do a byte by byte compare on the echoed data, and the entered data. If they do not compare, an invalid ASCII character should be entered in the string, the User Program should wait for the Error response, and then re-transmit the line.

3) The interface program is only invoked after a (CR) (ASCII X'0D') is received. Thus, the entire store command is buffered. Placing an invalid character in the store string will force an error. The entire store string is checked for validity before being executed. Therefore inputting an illegal character in the string will cause an error, and no data will be stored.

3. Compressed Store

a. Input

To issue a compressed store command, enter:

```
('S') + (' ') + (addr) + (':') + !(bytebyte)
```

This takes the bytes represented by the 'bytebytes' and stores them sequentially in NV-RAM starting at the given addr. There is no space preceding each 'bytebytes' inputted.

b. Output

(CRLF) + ('Ok') + (CRLF)

c. Notes:

See notes for Store.

4. Dump

a. Input

To issue a dump, enter:

```
('D') + (' ') + (addr) + (' ') + (addr) + (CR)
```

The bytes in NV-RAM, starting at the first addr and ending at the second addr, are converted to ASCII format and displayed on the terminal.

b. Output

The program will respond with:

```
!((':') + (addr) + (':') + !((' ') + (bytebyte)) + !(' ')

+ ('<') + (chksum) + ('>') + (CRLF)) + ('Ok') + (CRLF)
```

Addr is the address of the first byte in the line. Chksum is the checksum of the data bytes in the line.

c. Note

Because of the size of the internal data buffer, only 32 lines of data will be dumped as a maximum, regardless of how much data was requested. If more is asked for, the dump will simply terminate early.

5. Compressed Dump

a. Input

To issue a compressed dump, enter:

```
('C') + (' ') + (addr) + (' ') + (addr) + (CR)
```

The bytes in NV-RAM, starting at the first addr and ending at the second addr, are converted to ASCII format and displayed on the terminal.

b. Output

The program will respond with:

```
!((':') + (addr) + (':') + !(bytebyte) + ('<')

(chksum) + ('>') + (CRLF)) + ('Ok') + (CRLF)
```

Addr is the address of the first byte in the line. Chksum is the checksum of the data bytes in the line.

c. Notes

See note Dump.

Note: In all cases where input is unrecognizable or illegal, the response will be:

```
(CRLF) + ('Error') + (CRLF)
```

Note: In all cases when a store is attempted and other terminals are logged on, the response will be:

```
(CRLF) + ('Busy') + (CRLF)
```

E.5 Checksum

The checksum provided with the DUMP commands is defined by the following algorithm:

```
Always ignoring overflow:
begin
  for (each line dumped) begin
     checksum = 0;
     for (each of 1 to 16 bytes on the line) begin
       checksum = checksum + hex value of storage byte
     end
  output checksum
  end
end
```

So, in the line,

```
:1234: 1234 5678 9ABC DEF0 1234 5678 9ABC DEF0 <70>
```

the checksum is hex 70. And in the line,

```
:1234: 01A0 02                                 <A3>
```

the checksum is A3.

If a checksum problem is found, the dump command should be re-issued.

E.6 Examples

- Store

  A typical store command would look like:

  ```
  S 1234: 54F3 A534 78
  ```

  This would store the 5 bytes after the colon in memory starting at location 1234.

- Dump

  A typical dump command would look like this:

  ```
  D 1234 1245
  ```

  And a response might be:

  ```
  :1234: 1234 5678 9ABC DEF0 1234 5678 9ABC DEF0 <70>
  :1244: 1234                                    <46>
  ```

- Compressed Store

  A typical compressed store command would look like:

  ```
  S 1234:54F3A53478
  ```

This would store the 5 bytes after the colon in memory starting at location 1234.

- **Compressed Dump**

  A typical compressed dump command would look like this:

  ```
  C 1234 1245
  ```

  And a response might be:

  ```
  :1234:123456789ABCDEF0123456789ABCDEF0<70>
  :1244:1234<46>
  ```

Appendix F. ROM Data Base Organization

This section describes the IBM 7171 Data Base located in Read-Only Memory and lists the source code used to create the contents of the various tables.

The first 256 bytes of this ROM area are reserved for 128 addresses of tables used to define the terminals. A user may access a particular table by correctly coding the index. For example, to reference the reset character sequence (RCHS):

1. Convert the decimal index, 33, to hexadecimal X'21'.

2. Indicate it is an index by a one in the high order bit, X'8021'.

3. Byte reverse this value X'2180'.

Therefore the X'2180' in a terminal header will reference the reset sequence.

IBM 7171 ROM Data Base Organization

```
FIRST_ROM DS    0A
*
*            COMMON TERMINALS
               Table Name                  Decimal Index
                IBM3101                          0
                *            RESERVED            1
                TVI912                           2
                TVI920                           3
                TVI950                           4
                TVI950R                          5
                ADM31                            6
                ADM3A                            7
                VT100                            8
                DM1520                           9
                DM1521                          10
                DM3045                          11
                TYPETERM                        12
                HARDCOPY                        13
                PLOTTER                         14
*
*        COMMON TABLES / TABLE NAMES
                *                  SPARE        32
                RCHS                      INDEX 33
                QWERTY1                         34
                QWERTY2                         35
                QWERTY3                         36
                BITMATCH                        37
                TYPPFK                          38
                *                  SPARE        39
                FUNCPFK                         40
                FUNBPFK                         41
                FUNCX                           42
                FUNCA                           43
                RSETCHAR                        44
                SETCHAR                         45
                SETMTCH                         46
                COMSET                          47
                SGR                             48
                DM3SGR                          49
                ADM31ATR                        50
                ADM31MOD                        51
                *            RESERVED           52
                IBM3101I                        53
                IBM3101_CSS1                    54
                IBMESC                          55
                IBMESC2                         56
                IBMESC3                         57
                IBMSET                          58
                IBMSET2                         59
                DM1520I                         60
                DM1520_CSS1                     61
                DMESC                           62
                DMSET                           63
                DM3045I                         64
                DM3045_CSS1                     65
                D3ESC                           66
                D3ESC2                          67
                D3SET                           68
                D3SET2                          69
                DM3PFKS                         70
                VT100I                          71
                VT100_CSS1                      72
                VTE                             73
```

VTEE		74
VTEEO		75
VTEO		76
VTEOP		77
VTEOPE		78
VTEOPEO		79
VTSET		80
VTSET2		81
VTSET3		82
VTPFKS		83
*	RESERVED	84
ADM3AI		85
ADM3A_CSS1		86
A3AESC		87
ADM31I		88
ADM31_CSS1		89
ADMNORM		90
A31ESC		91
A31SET		92
A31ESET		93
A31EESC		94
TVIDI		95
TVDSOH		96
TVSET1		97
TVSET2		98
TVII		99
TVI950_CSS1		100
TVISOH		101
TVIESC		102
TVISET		103
TVIPFK1		104
TVIPFK2		105
TPTMI		106
TYPETERM_CSS1		107
TPTMESC		108
HCPYI		109
HARDCOPY_CSS1		110
HCPYESC		111
TTRAN		112
HTRAN		113
HR77		114
HRDA		115
HW77		116
HWDA		117
MSGADDR		118
TTRAN7E		119

```
                 HOST TRANSLATE TABLE POINTERS
HTRAN     DS     0A
          DC     A(HR77)             HOST READ 3277
          DC     A(HRDA)             HOST READ APL
          DC     A(HW77)             HOST WRITE
          DC     A(HWDA)             HOST WRITE APL
          DC     A(HW3278)           HOST WRITE 3278
          DC     A(HW3278AT)
          DC     A(HW3278ES)
          DC     A(HR3278)           HOST READ 3278
          DC     A(HR3278AT)
          DC     A(HR3278ES)


          See Appendix A for the
      HOST TRANSLATE TABLES for 3277 and 3278:
          HR77
          HRDA
          HW77
          HWDA
          HW3278
          HW3278AT
          HW3278ES
          HR3278
          HR3278AT
          HR3278ES

          See Appendix A for the
         TERMINAL TRANSLATE TABLES
TTRAN     DS     0A
          NORMAL (NON-APL) ASCII TABLE
          NORMAL ASCII INPUT TRANSLATE TABLE
          TYPEWRITER PAIRED APL TABLE
          TYPEWRITER PAIRED APL INPUT TRANSLATE TABLE
          BIT PAIRED APL TABLE
          BIT PAIRED APL INPUT TRANSLATE TABLE
***********************************************************
```

```
             ALTERNATE KEYBOARD
*************************************************************
* REARRANGE GRAPHIC KEYS FOR DEVORAK KEYBOARD               *
*************************************************************
*              0 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'00000000000000000000000000000000' 0
         DC    X'00000000000000000000000000000000' 1
         DC    X'202122232425262728292A2B772D767A' 2
         DC    X'303132333435363738395373573D565A' 3
         DC    X'4041584A453E5549444348544E4D4252' 4
         DC    X'4C3F504F59474B3C51463A5B5C5D5E5F' 5
         DC    X'6061786A652E7569646368746E6D6272' 6
         DC    X'6C2F706F79676B2C71663B7B7C7D7E00' 7
*              0 1 2 3 4 5 6 7 8 9 A B C D E F
*************************************************************
*
*        TERMINAL RESET CONTROL CHARACTER SEQUENCES
*
*        ****************************************************
*        * PERFORM RESET FUNCTIONS                          *
*        *     RESET INTRODUCER (X'FF' = NONE IF NOT USED)  *
*        *     MASTER RESET                                 *
*        *     CHARACTER ERROR RESET                        *
*        *     KEYBOARD UNLOCK (3270 "RESET")               *
*        *     TYPEAHEAD PURGE                              *
*        *     PACING START                                 *
*        *     PACING STOP                                  *
*        *     OPERATOR TOGGLE                              *
*        ****************************************************
*
RCHS     CSS NONE,BEL,DC2,DC4,CAN,XON,XOFF,ETB
* CTRL            G   R   T   X   Q   S   W
*
*************************************************************
*
*        SET GRAPHIC RENDITION CODE SEQUENCES
*
SGR      DC    X'00000100',X'01000001',X'0001' ANSI X3.64
DM3SGR   DC    X'4F4F4E4F',X'4E4E4E4E',X'4E4E' DM3045
ADM31ATR DC    X'30303430',X'34303030',X'3030' ADM31 ATTR BYTE
ADM31MOD DC    X'28292829',X'28282828',X'2828' ADM31 MODE
         DC    X'00000100',X'01373733',X'3631' (RESERVED)
*
*************************************************************
*
*        COMMON PROGRAM FUNCTION KEY TABLES
*
*        ****************************************************
*        * THERE IS  A COMMON SET OF TWO KEY FUNCTION       *
*        * DEFINITIONS DEFINED ACROSS MOST TERMINALS.       *
*        * IT DEFINES A FUNCTION INTRODUCER KEY             *
*        *    ERASE EOS ON THE IBM 3101                     *
*        *    ESC   ON MOST OTHER TERMINALS                 *
*        *    FUNCT ON THE ADM FAMILY AND ITS FOLLOWERS     *
*        * THE SECOND KEY DEFINES THE NATURE OF THE REQUEST *
*        * ON TYPEWRITER PAIRED TERMINALS, THEY ARE         *
*        *    PFK1-12 FOR THE TOP KEYBOARD ROW 1234567890-= *
*        *    PFK13-24 FOR THIS ROW SHIFTED    !@#$%¬&*() + *
*        *    PFK13-24 FOR THE NEXT ROW        QWERTYUIOP   *
*        *    PFK25-36 FOR                     ASDFGHJKL;'  *
*        *    PA1-3    FOR THE LOWER RIGHT     ,./          *
*        ****************************************************
*
*        ****************************************************
*        * UNSHIFTED QWERTY TYPEWRITER PAIRED TERMINAL      *
*        * (ANSI X4.14-1971) ROWS E TO C COLS 00 TO 12      *
```

```
*        ****************************************************
QWERTY1      CSS  '1234567890-=',XX
             CSS  'qwertyuiop[',BSLASH,XX
             CSS  'asdfghjkl;',QUOTE,LBRACE
*        ****************************************************
*        * SHIFTED QWERTY TYPEWRITER PAIRED TOP ROW         *
*        ****************************************************
QWERTY2      CSS  '!@#$%',UPARROW,AND,'*()_+'
*        ****************************************************
*        * SHIFTED QWERTY TYPEWRITER PAIRED MIDDLE ROWS     *
*        ****************************************************
QWERTY3      CSS  'QWERTYUIOP]',BAR,XX
             CSS  'ASDFGHJKL;',QUOTE,RBRACE
*        ****************************************************
*        * ANSI X4.14 BIT PAIRED DIFFERENCES FROM TYPEPAIR  *
*        ****************************************************
BITMATCH     CSS  DQUOTE,14,AND,18,ACCENT,19,LPAREN,20,RPAREN,21,
               MINUS,11,UPARROW,12,EQUAL,23,TILDE,24,XX
             CSS  AT,23,LBRACK,24,COLON,35,RBRACK,36
*
*        ****************************************************
*        * ANSI STANDARD TYPEPAIRED ARRANGEMENT             *
*        ****************************************************
TYPPFK       NODE INDEX,GOTO,QWERTY1,PFK1
             NODE NULL,CALL,ATTN
*
             NODE INDEX,GOTO,QWERTY2,PFK13
             NODE NULL,CALL,ATTN
*
             NODE INDEX,GOTO,QWERTY3,PFK13
             NODE NULL,CALL,ATTN
*
             NODE COMMA,CALL,PA1
             NODE LESS,CALL,PA1
             NODE PERIOD,CALL,PA2
             NODE GREATER,CALL,PA2
             NODE SLASH,CALL,PA3
             NODE QUESTION,CALL,PA3
*
             NODE SEQENDER,GOTO,*
*
*        ****************************************************
*        * TYPE PAIRED FUNC TABLE                           *
*        ****************************************************
FUNCPFK      NODE INDEX,GOTO,QWERTY1,PFK1
             NODE NULL,GOTO,FUNCA
             NODE INDEX,GOTO,QWERTY2,PFK13
             NODE NULL,GOTO,FUNCA
             NODE INDEX,GOTO,QWERTY3,PFK13
             NODE NULL,GOTO,FUNCA
*
             NODE COMMA,GOTO,FUNCA,PA1
             NODE LESS,GOTO,FUNCA,PA1
             NODE PERIOD,GOTO,FUNCA,PA2
             NODE GREATER,GOTO,FUNCA,PA2
             NODE SLASH,GOTO,FUNCA,PA3
             NODE QUESTION,GOTO,FUNCA,PA3
*
             NODE SEQENDER,GOTO,*
*
*        ****************************************************
*        * BIT PAIRED INTRODUCER SEQUENCE                   *
*        ****************************************************
FUNBPFK      NODE MATCH,GOTO,BITMATCH,ENTER
```

```
            NODE NULL,GOTO,FUNCA
*
            NODE OTHERS,GOTO,FUNCPFK
*
*        *****************************************************
*        * FUNCTION SEQUENCE TERMINATOR FOR LOCAL KEYS       *
*        *****************************************************
FUNCX       NODE CR,CALL,DEFER
            NODE SEQENDER,GOTO,*
*        *****************************************************
*        * FUNCTION SEQUENCE TERMINATOR FOR ATTN KEYS        *
*        *****************************************************
FUNCA       NODE CR,CALL,ATTN
            NODE SEQENDER,GOTO,*
*
*
*
***************************************************************
*
*                  COMMON SETUP TABLES
*
*                  0123456789ABCDEF   BIT POSITION IN FLAG WORD
RSETCHAR    CSS  'ZNECV   DI      '
SETCHAR     CSS  'znecv   di      '
SETMTCH     CSS  LCP,INPCON,P,INPCOFF,LCA,INAPLON,A,INAPLEND,
              LCM,INAPLOFF,HT,INSETTAB,DEL,INDELTAB,
              PERIOD,INDISC,COMMA,ININIT
COMSET      NODE INDEX,GOTO,RSETCHAR
            NODE NULL,CALL,RESET
            NODE INDEX,GOTO,SETCHAR
            NODE NULL,CALL,SET
            NODE MATCH,GOTO,SETMTCH
            NODE NULL,CALL,DEFER
            NODE LCQ,CALL,ALTK,PARM=1
            NODE Q,CALL,ALTK
            NODE SEQENDER,GOTO,*
*
***************************************************************
```

```
**************************************************************
*
*        TERMINAL DEFINITION TABLES (TDT'S)
*
*
************************************************************ *
* IBM3101 TABLE                                              *
************************************************************ *
*
IBM3101      TERM IBM3101I
             CSS ESC,Y,(BINY),(BINX) REPOSITION
             CSS ESC,I             ERASE EOL
             CSS ESC,W             LOCAL PRINT
             CSS BEL               TONE
             CSS BS                CURSOR LEFT
             CSS ESC,C             CURSOR RIGHT
             CSS ,                 -- NOT USED --
             CSS ,                 -- NOT USED --
             CSS ,                 SIGNAL INSERT MODE
             CSS ,                 SIGNAL END INSERT MODE
             CSS ,                 DISCONNECT
             CSS ESC,L             CLEAR
             CSS ESC,L             TERMINAL INITIALIZE
             CSS COLON             ILLEGAL ASCII CHAR
             CSS ,                 ILLEGAL APL CHAR
             CSS ,                 APL CHARS ON
             CSS ,                 APL CHARS OFF
             CSS ,                 DISPLAY MODE
             CSS ,                 -- NOT USED --
*        *****************************************************
*        * 3101 PRIMARY INPUT PARSE TABLE                    *
*        *****************************************************
IBM3101I     NODE ESC,GOTO,IBMESC
             NODE HT,CALL,TAB
             NODE BS,CALL,ENTER
             NODE CR,CALL,NL
             NODE DEL,CALL,DELETE
             NODE SYN,CALL,RESHOW
             NODE VT,CALL,DPFM,DUP          DUP CHARACTER
             NODE FF,CALL,DPFM,FM           FM CHARACTER
             NODE SEQENDER,GOTO,*
*        *****************************************************
*        * 3101 ESCAPE FOLLOWON TABLE                        *
*        *****************************************************
IBMESC       NODE J,GOTO,IBMESC2          EEOS
             NODE HT,CALL,COLTAB
             NODE CR,CALL,COLBAK
             NODE A,CALL,UP               CURSOR UP
             NODE B,CALL,DOWN             CURSOR DOWN
             NODE C,CALL,RIGHT            CURSOR RIGHT
             NODE D,CALL,LEFT             CURSOR LEFT
             NODE H,CALL,HOME             HOME
             NODE I,CALL,EEOF             ERASE EOF/EOL
             NODE K,CALL,CURSEL           ERASE INPUT
             NODE L,CALL,CLEAR            CLEAR
             NODE W,CALL,LPRT             PRINT
             NODE SEQENDER,GOTO,*
*        *****************************************************
*        * 3101 FUNCTION TABLE (FOLLOWING ERASE EOS KEY)     *
*        *****************************************************
IBMESC2      NODE ESC,GOTO,IBMESC3
             NODE HT,CALL,BTAB            FUNCT TAB
             NODE LCZ,CALL,PA1
             NODE LCX,CALL,PA2
             NODE LCC,CALL,PA3
```

```
         NODE Z,CALL,PA1
         NODE X,CALL,PA2
         NODE C,CALL,PA3
         NODE DEL,CALL,INSRT          FUNCT DEL
         NODE ACCENT,GOTO,IBMSET
         NODE OTHERS,GOTO,TYPPFK
*       ******************************************************
*       * 3101 EEOS-ESC-SOMETHING                             *
*       ******************************************************
IBMESC3  NODE A,CALL,INDENT           FUNCT-UP
         NODE B,CALL,UNDENT           FUNCT-DOWN
         NODE C,CALL,COLTAB           FUNCT-RIGHT
         NODE D,CALL,COLBAK           FUNCT-LEFT
         NODE K,CALL,EINP             FUNCT-ERASE INPUT
         NODE W,CALL,TREQ             FUNCT-PRINT
         NODE SEQENDER,GOTO,*
*       ******************************************************
*       * 3101 SETUP SEQUENCE (FUNCT-ACCENT-SOMETHING)        *
*       ******************************************************
IBMSET   NODE ESC,GOTO,IBMSET2
         NODE CR,CALL,SETMRG           NL
         NODE OTHERS,GOTO,COMSET
IBMSET2  NODE H,CALL,SETHOM
         NODE L,CALL,CLRTAB
         NODE SEQENDER,GOTO,*
*
```

```
************************************************************ *
* DATAMEDIA 1520 TABLE                                       *
************************************************************ *
DM1520       TERM DM1520I
             CSS RS,(BINX),(BINY)      POSITION
             CSS GS                    ERASE EOL
             CSS VT                    LOCAL PRINT
             CSS BEL                   TONE
             CSS BS                    CURSOR LEFT
             CSS FS                    CURSOR RIGHT
             CSS ,                     -- NOT USED --
             CSS ,                     -- NOT USED --
             CSS DEL                   SIGNAL INSERT MODE
             CSS DEL                   SIGNAL END INSERT MODE
             CSS EM                    DISCONNECT
             CSS FF                    CLEAR
             CSS FF                    TERMINAL INITIALIZE
             CSS COLON                 ILLEGAL ASCII CHAR
             CSS LCN,BS,LCZ            ILLEGAL APL CHAR
             CSS ,                     APL CHARS ON
             CSS ,                     APL CHARS OFF
             CSS ,                     DISPLAY MODE
             CSS ,                     -- NOT USED --
*        ***************************************************
*        * DM1520 INPUT PARSE TABLE                        *
*        ***************************************************
DM1520I      NODE ESC,GOTO,DMESC       TYPE PAIRED ENTRY
             NODE SI,CALL,NL           PRINT OFF
             NODE FF,CALL,CLEAR        CLEAR
             NODE BS,CALL,LEFT
             NODE FS,CALL,RIGHT
             NODE US,CALL,UP
             NODE LF,CALL,DOWN
             NODE GS,CALL,INSRT        ERAS EOL
             NODE DEL,CALL,DELETE      DEL
             NODE HT,CALL,TAB          TAB
             NODE SO,CALL,COLTAB       PRINT
             NODE CR,CALL,ENTER        RETURN
             NODE VT,CALL,EEOF         ERAS EOS
             NODE EM,CALL,HOME         HOME
             NODE SYN,CALL,RESHOW      CTRL-V
             NODE SEQENDER,GOTO,*
*        ***************************************************
*        * DATAMEDIA 1520 ESCAPE SEQUENCE                  *
*        ***************************************************
DMESC        NODE HT,CALL,BTAB         FUNCT TAB
             NODE BS,CALL,BTAB         FUNCT LEFT
             NODE FS,CALL,TAB          FUNCT RIGHT
             NODE US,CALL,INDENT       FUNCT UP
             NODE LF,CALL,UNDENT       FUNCT DOWN
             NODE SO,CALL,COLBAK       FUNCT PRINT (COLTAB)
             NODE ACCENT,GOTO,DMSET    FUNCT ACCENT
             NODE OTHERS,GOTO,TYPPFK
*        ***************************************************
*        * DATAMEDIA 1520 SETUP TABLE                      *
*        ***************************************************
DMSET        NODE SO,CALL,SETTAB
             NODE FF,CALL,CLRTAB
             NODE SI,CALL,SETMRG
             NODE EM,CALL,SETHOM
             NODE OTHERS,GOTO,COMSET
************************************************************ *
* DATAMEDIA 1521 TABLE                                       *
************************************************************ *
DM1521       TERM DM1520I,DM1520
```

```
****************************************************************
* DATAMEDIA 3000 SERIES TERMINAL TABLE                         *
****************************************************************
DM3045       TERM DM3045I,FLAGS=X'4C00',SGR=DM3SGR
             CSS ESC,Y,(BINX),(BINY)    REPOSITION
             CSS ESC,K                  ERASE EOL
             CSS ,                      LOCAL PRINT
             CSS BEL                    TONE
             CSS BS                     CURSOR LEFT
             CSS ESC,C                  CURSOR RIGHT
             CSS ,                      -- NOT USED --
             CSS ,                      -- NOT USED --
             CSS ,                      SIGNAL INSERT MODE
             CSS ,                      SIGNAL END INSERT MODE
             CSS ,                      DISCONNECT
             CSS ESC,H,ESC,J            CLEAR
             CSS ESC,M,SI               TERMINAL INITIALIZE
             CSS COLON                  ILLEGAL ASCII CHAR
             CSS LCN,BS,LCZ             ILLEGAL APL CHAR
             CSS SO                     APL CHARS ON
             CSS SI                     APL CHARS OFF
             CSS ESC,(BINFLD)           DISPLAY MODE
             CSS ,                      -- NOT USED --
*        *******************************************************
*        * DM3045 INPUT PARSE TABLES                           *
*        *******************************************************
DM3045I      NODE ESC,GOTO,D3ESC
             NODE CR,CALL,ENTER         RETURN
             NODE BS,CALL,LEFT
             NODE LF,CALL,DOWN
             NODE DEL,CALL,DELETE
             NODE HT,CALL,TAB
             NODE SYN,CALL,RESHOW       CTRL-V
             NODE SO,CALL,APLON         CTRL-N
             NODE SI,CALL,APLOFF        CTRL-O
             NODE SEQENDER,GOTO,*
*        *******************************************************
*        * DM3045 ESCAPE TABLE                                 *
*        *******************************************************
D3ESC        NODE ESC,GOTO,D3ESC2
             NODE RBRACK,CALL,NL        PRINT
             NODE A,CALL,UP
             NODE B,CALL,LEFT           SHIFT LEFT
             NODE C,CALL,RIGHT
             NODE H,CALL,HOME           HOME
             NODE J,CALL,EEOF           ERAS EOS
             NODE Q,CALL,UP             SHIFT CURSOR UP
             NODE R,CALL,RIGHT          SHIFT CURSOR RIGHT
             NODE S,CALL,DOWN           SHIFT CURSOR DOWN
             NODE V,CALL,COLTAB         ROLL
             NODE W,CALL,COLBAK         SHIFT ROLL
             NODE M,CALL,CLEAR          MR
             NODE P,CALL,INSRT          INS CHAR
             NODE HT,CALL,BTAB          TAB
             NODE BS,CALL,BTAB          FUNCT  LEFT
             NODE LF,CALL,UNDENT        FUNCT  DOWN
             NODE INDEX,GOTO,DM3PFKS,PFK1
             NODE NULL,GOTO,FUNCA
             NODE RBRACE,CALL,APLEND
             NODE ACCENT,GOTO,D3SET     FUNCT ACCENT
             NODE TILDE,CALL,PFK26      PFK26 ALTERNATE
             NODE OTHERS,GOTO,TYPPFK    (EXCLUDE FUNCT-QWERTY)
*        *******************************************************
*        * ESC-ESC SEQUENCE                                    *
*        *******************************************************
D3ESC2       NODE C,CALL,TAB            FUNCT  RIGHT
```

```
         NODE A,CALL,INDENT         FUNCT  UP
         NODE V,CALL,COLBAK         FUNCT  COLTAB
         NODE INDEX,GOTO,DM3PFKS,PFK13
         NODE NULL,GOTO,FUNCA
         NODE SEQENDER,GOTO,*
*       ******************************************************
*       * SETUP MODE                                         *
*       ******************************************************
D3SET    NODE ESC,GOTO,D3SET2
         NODE OTHERS,GOTO,COMSET
D3SET2   NODE RBRACK,CALL,SETMRG
         NODE M,CALL,CLRTAB
         NODE H,CALL,SETHOM
         NODE SEQENDER,GOTO,*
DM3PFKS  CSS  'pqrstuvwxyLT'
*
```

```
************************************************************ *
* VT100 TERMINAL TABLE                                       *
************************************************************ *
VT100        TERM VT100I,ORIGIN=1,FLAGS=X'0C00'
             CSS ESC,LBRACK,(CHARY),SEMI,(CHARX),H  REPOSITION
             CSS ESC,LBRACK,K             ERASE EOL
             CSS ,                        LOCAL PRINT
             CSS BEL                      TONE
             CSS BS                       CURSOR LEFT
             CSS ESC,LBRACK,C             CURSOR RIGHT
             CSS ,                        -- NOT USED --
             CSS ,                        -- NOT USED --
             CSS ESC,LBRACK,@1,LCQ        SIGNAL INSERT MODE
             CSS ESC,LBRACK,@0,LCQ        SIGNAL END INSERT MODE
             CSS ,                        DISCONNECT
             CSS ESC,LBRACK,H,ESC,LBRACK,J   CLEAR
             CSS '¢=¢<¢[H¢[J¢[?1h¢[?3;6;7l¢[20l¢[0q¢[0m'
*                                         TERMINAL INITIALIZE
             CSS COLON                    ILLEGAL ASCII CHAR
             CSS ,                        ILLEGAL APL CHAR
             CSS ,                        APL CHARS ON
             CSS ,                        APL CHARS OFF
             CSS ESC,LBRACK,(CHARFLD),LCM DISPLAY MODE
             CSS ,                        -- NOT USED --
*       ******************************************************
*       * VT100 FIRST INPUT PARSE TABLE                      *
*       ******************************************************
VT100I       NODE ESC,GOTO,VTE
             NODE DEL,CALL,DELETE           "DELETE"
             NODE HT,CALL,COLTAB            "TAB"
             NODE CR,CALL,ENTER             "RETURN"
             NODE LF,CALL,NL                "LINE FEED"
             NODE BS,CALL,HOME              "BACK SPACE"
             NODE SYN,CALL,RESHOW           "CTRL"-V
             NODE SEQENDER,GOTO,*
*       ******************************************************
*       * ESCAPE SEQUENCE                   "ESC"-           *
*       ******************************************************
VTE          NODE ESC,GOTO,VTEE
             NODE O,GOTO,VTEO
             NODE LBRACK,CALL,PFK23
             NODE RBRACK,CALL,PFK24
             NODE LBRACE,CALL,PFK23
             NODE RBRACE,CALL,PFK24
             NODE CR,CALL,PFK36             "RETURN"
             NODE HT,CALL,COLBAK            "TAB"
             NODE DEL,CALL,EEOF             "DELETE"
             NODE ACCENT,GOTO,VTSET
             NODE OTHERS,GOTO,TYPPFK
             NODE SEQENDER,GOTO,*
*       ******************************************************
*       * ESC-ESC SEQUENCE                                   *
*       ******************************************************
VTEE         NODE O,GOTO,VTEEO
             NODE SEQENDER,GOTO,*
*       ******************************************************
*       * ESC- CURSOR MOVEMENT SEQUENCE                      *
*       ******************************************************
VTEEO        NODE D,CALL,BTAB               LEFT ARROW
             NODE C,CALL,TAB                RIGHT ARROW
             NODE A,CALL,INDENT             UP ARROW
             NODE B,CALL,UNDENT             DOWN ARROW
             NODE INDEX,GOTO,VTPFKS,PFK25
             NODE NULL,CALL,ATTN
             NODE SEQENDER,GOTO,*
*       ******************************************************
```

```
*       * ESC,O SEQUENCE      (KEYPAD AND CURSOR KEYS)        *
*       *******************************************************
VTEO        NODE M,CALL,CLEAR               "ENTER"
            NODE A,CALL,UP                  UP ARROW
            NODE B,CALL,DOWN                DOWN ARROW
            NODE C,CALL,RIGHT               RIGHT ARROW
            NODE D,CALL,LEFT                LEFT ARROW
            NODE S,CALL,PA1                 "PF 4"
            NODE LCM,CALL,PA2               "-"
            NODE LCL,CALL,PA3               ","
            NODE LCN,CALL,INSRT             PERIOD
            NODE INDEX,GOTO,VTPFKS,PFK1
            NODE NULL,CALL,ATTN
            NODE LCP,GOTO,VTEOP             "0" PREFIX 'SHIFTED' PFK'S
            NODE SEQENDER,GOTO,*
*       *******************************************************
*       * ESC,O,P SEQUENCE FOR "SHIFTED" PFK'S (13-24)        *
*       *******************************************************
VTEOP       NODE ESC,GOTO,VTEOPE          BEGIN SECOND KEY
            NODE SEQENDER,GOTO,*
*       *******************************************************
*       * ESC,O,P,ESC  "SHIFTED" PFK SEQUENCE                 *
*       *******************************************************
VTEOPE      NODE O,GOTO,VTEOPEO           MORE OF "SHIFTED" PFK'S
            NODE SEQENDER,GOTO,*
*       *******************************************************
*       * ESC,O,P,ESC,O "SHIFTED" PFK'S COME HERE             *
*       *******************************************************
VTEOPEO     NODE INDEX,GOTO,VTPFKS,PFK13
            NODE NULL,CALL,ATTN
            NODE SEQENDER,GOTO,*
*       *******************************************************
*       * ESC,ACCENT SETUP REQUEST                            *
*       *******************************************************
VTSET       NODE LF,CALL,SETMRG           "LINE FEED"
            NODE ESC,GOTO,VTSET2
            NODE BS,CALL,SETHOM           "BACK SPACE"
            NODE OTHERS,GOTO,COMSET
VTSET2      NODE O,GOTO,VTSET3
            NODE SEQENDER,GOTO,*
VTSET3      NODE M,CALL,CLRTAB            "ENTER"
            NODE SEQENDER,GOTO,*
VTPFKS      CSS  'PQRwxytuvqrs'
*
```

```
**************************************************************
* ADM3A TERMINAL TABLE                                       *
**************************************************************
ADM3A        TERM ADM3AI
             CSS ESC,EQUAL,(BINY),(BINX)  REPOSITION
             CSS ,                        ERASE EOL
             CSS ,                        LOCAL PRINT
             CSS BEL                      TONE
             CSS BS                       CURSOR LEFT
             CSS ,                        CURSOR RIGHT
             CSS ,                        -- NOT USED --
             CSS ,                        -- NOT USED --
             CSS ,                        SIGNAL INSERT MODE
             CSS ,                        SIGNAL END INSERT MODE
             CSS ,                        DISCONNECT
             CSS SUB                      CLEAR
             CSS SUB                      TERMINAL INITIALIZE
             CSS COLON                    ILLEGAL ASCII CHAR
             CSS ,                        ILLEGAL APL CHAR
             CSS ,                        APL CHARS ON
             CSS ,                        APL CHARS OFF
             CSS ,                        DISPLAY MODE
             CSS ,                        -- NOT USED --
*        *****************************************************
*        * INPUT PARSE TABLE                                 *
*        *****************************************************
ADM3AI       NODE ESC,GOTO,A3AESC            ESCAPE SEQUENCE START
             NODE BS,CALL,LEFT               H CURSOR LEFT
             NODE FF,CALL,RIGHT              L CURSOR RIGHT
             NODE VT,CALL,UP                 K CURSOR UP
             NODE LF,CALL,DOWN               J CURSOR DOWN
             NODE EM,CALL,INSRT              Y TOGGLE INSERT MODE
             NODE DEL,CALL,DELETE              DELETE
             NODE HT,CALL,TAB                I TAB
             NODE SI,CALL,BTAB               O BACKTAB
             NODE NAK,CALL,NL                U NEWLINE
             NODE CR,CALL,ENTER                ENTER
             NODE SUB,CALL,CLEAR             Z CLEAR
             NODE DLE,CALL,EEOF              P ERASE EOF
             NODE RS,CALL,HOME                 HOME
             NODE SYN,CALL,RESHOW            V REDISPLAY SCREEN
             NODE SEQENDER,GOTO,*
*        *****************************************************
*        * ADM3A ESC SEQUENCE                                *
*        *****************************************************
A3AESC       NODE HT,CALL,COLTAB
             NODE SI,CALL,COLBAK
             NODE FF,CALL,INDENT
             NODE BS,CALL,UNDENT
             NODE ACCENT,GOTO,A3ASET
             NODE OTHERS,GOTO,TYPPFK
*        *****************************************************
*        * ADM3A SETUP SEQUENCE                              *
*        *****************************************************
A3ASET       NODE RS,CALL,SETHOM
             NODE NAK,CALL,SETMRG
             NODE SUB,CALL,CLRTAB
             NODE OTHERS,GOTO,COMSET
*
```

```
*********************************************************** *
* ADM31 TERMINAL TABLE                                      *
* (ALSO RELATED TO TELEVIDEO, ZENTEC, AND OTHER TERMINALS)  *
*********************************************************** *
ADM31        TERM ADM31I,FLAGS=X'0000',SGR=ADM31ATR
             CSS ESC,EQUAL,(BINY),(BINX)  REPOSITION
             CSS ESC,T                    ERASE EOL
             CSS ,                        LOCAL PRINT
             CSS BEL                      TONE
             CSS BS                       CURSOR LEFT
             CSS ,                        CURSOR RIGHT
             CSS ,                        -- NOT USED -
             CSS ,                        -- NOT USED --
             CSS ,                        SIGNAL INSERT MODE
             CSS ,                        SIGNAL END INSERT MODE
             CSS ,                        DISCONNECT
             CSS RS,ESC,Y                 CLEAR
             CSS ESC,C,ESC,D,F,ESC,QUOTE,ESC,LCG,RS,ESC,Y
*                                         TERMINAL INITIALIZE
             CSS COLON                    ILLEGAL ASCII CHAR
             CSS ,                        ILLEGAL APL CHAR
             CSS ,                        APL CHARS ON
             CSS ,                        APL CHARS OFF
             CSS ESC,G,(BINFLD)           DISPLAY MODE
             CSS ,                        -- NOT USED --
*       *****************************************************
*       * INPUT PARSE TABLE                                 *
*       *****************************************************
ADM31I       NODE SOH,GOTO,FUNBPFK        "FUNCT" KEY INITIAL CHAR
             NODE ESC,GOTO,A31ESC         ESCAPE SEQUENCE
             NODE US,CALL,NL              "NEW LINE" KEY
ADMNORM      NODE BS,CALL,LEFT
             NODE FF,CALL,RIGHT
             NODE VT,CALL,UP
             NODE LF,CALL,DOWN
             NODE DEL,CALL,DELETE
             NODE HT,CALL,COLTAB
             NODE CR,CALL,ENTER
             NODE RS,CALL,HOME            "HOME" KEY
             NODE SYN,CALL,RESHOW         CTRL-V
             NODE SEQENDER,GOTO,*
*       *****************************************************
*       * ESCAPE SEQUENCE TABLE                             *
*       *****************************************************
A31ESC       NODE ESC,GOTO,A31EESC
             NODE Q,CALL,NULL             "CHAR INSERT"
             NODE W,CALL,CLEAR            "CHAR DELETE"
             NODE E,CALL,INSRT            "LINE INSERT"
             NODE R,CALL,EEOF             "LINE DELETE"
             NODE T,CALL,TAB              "LINE ERASE"
             NODE Y,CALL,INDENT           "PAGE ERASE"
             NODE @5,CALL,CURSEL          "SEND PAGE"
             NODE I,CALL,COLBAK           "BACKTAB"
             NODE ACCENT,GOTO,A31SET
             NODE SEQENDER,GOTO,*
*       *************************************************** *
*       * ADM31 SETUP SEQUENCE                              *
*       *************************************************** *
A31SET       NODE RS,CALL,SETHOM
             NODE US,CALL,SETMRG
             NODE ESC,GOTO,A31ESET
             NODE OTHERS,GOTO,COMSET
A31ESET      NODE W,CALL,CLRTAB
             NODE SEQENDER,GOTO,*
*       *****************************************************
```

```
*         * ESCAPE FOLLOWED BY A FUNCTION KEY LOGICALLY         *
*         * SHIFTS IT                                           *
*         *******************************************************
A31EESC       NODE Q,CALL,DPFM,DUP          "CHAR INSERT" = DUP
              NODE W,CALL,DPFM,FM           "CHAR DELETE" = FM
              NODE E,CALL,LPRT              "LINE INSERT"
              NODE R,CALL,EINP              "LINE DELETE"
              NODE T,CALL,BTAB              "LINE ERASE"
              NODE Y,CALL,UNDENT            "PAGE ERASE"
              NODE @5,CALL,TREQ             "SEND PAGE"
              NODE @7,CALL,TREQ             "SEND"
              NODE SEQENDER,GOTO,*
*
```

```
************************************************************ *
* TELEVIDEO 912 TERMINAL                                     *
************************************************************ *
TVI912      TERM TVIDI,ADM31
TVIDI       NODE SOH,GOTO,TVDSOH         ESCAPE INTRODUCER
            NODE ESC,CALL,INSRT
            NODE SUB,CALL,CLEAR
            NODE RS,CALL,NL              "HOME" (NO GOOD NEWLINE)
            NODE OTHERS,GOTO,ADMNORM     REST LIKE ADM31
*       *****************************************************
*       * FUNCT  SEQUENCE TABLE                             *
*       *****************************************************
TVDSOH      NODE HT,GOTO,FUNCX,INCOLBAK
            NODE DEL,GOTO,FUNCX,INEEOF ERASE EOF
            NODE BS,GOTO,FUNCX,INBTAB       LEFT
            NODE FF,GOTO,FUNCX,INTAB        RIGHT
            NODE VT,GOTO,FUNCX,ININDENT     UP
            NODE LF,GOTO,FUNCX,INUNDENT     DOWN
            NODE ACCENT,GOTO,TVSET1
            NODE RS,GOTO,FUNCX,INHOME
            NODE OTHERS,GOTO,FUNCPFK
*       *****************************************************
*       * FUNCTION SEQUENCE TERMINATOR FOR SETUP INTRODUCER  *
*       *****************************************************
TVSET1      NODE CR,GOTO,TVSET2
            NODE SEQENDER,GOTO,*
TVSET2      NODE ESC,CALL,SETHOM
            NODE SUB,CALL,CLRTAB
            NODE RS,CALL,SETMRG
            NODE OTHERS,GOTO,COMSET
*
************************************************************ *
* TELEVIDEO 920 TERMINAL                                     *
************************************************************ *
TVI920      TERM TVIDI,ADM31
*
```

```
*********************************************************** *
* TELEVIDEO 950 TABLE                                       *
*********************************************************** *
* SHARED OUTPUT CONTROL STRINGS, REFERENCED BELOW
*
TVICS1        CSS ESC,EQUAL,(BINY),(BINX) POSITION
TVICS2        CSS ESC,T                   ERASE EOL
TVICS3        CSS ,                       LOCAL PRINT
TVICS4        CSS BEL                     TONE
TVICS5        CSS BS                      CURSOR LEFT
TVICS6        CSS FF                      CURSOR RIGHT
TVICS7        CSS ,                       -- NOT USED --
TVICS8        CSS ,                       -- NOT USED --
TVICS9        CSS ESC,LCF,'INSERT MODE',CR SIGNAL INSERT MODE
TVICS10       CSS ESC,LCF,CR              SIGNAL END INSERT MODE
TVICS11       CSS ,                       DISCONNECT
TVICS12       CSS ESC,ASTERISK,(DELAY)    CLEAR
TVICS13       CSS SI,'¢O¢C¢DF¢%¢"',DC4,ESC,QUOTE,'¢G¢F¢G0',CR,'¢F',
                CR,(DELAY),'¢017¢026',XX
              CSS RS,SUB,'¢n¢a¢O¢X¢l¢r',(DELAY)
*                                         TERMINAL INITIALIZE
TVICS14       CSS COLON                   ILLEGAL ASCII CHAR
TVICS15       CSS ,                       ILLEGAL APL CHAR
TVICS16       CSS ,                       APL CHARS ON
TVICS17       CSS ,                       APL CHARS OFF
TVICS18A      CSS ESC,G,(BINFLD)          DISPLAY MODE (ATR - TVI950R)
TVICS18B      CSS ESC,(BINFLD)            DISPLAY MODE (MOD - TVI950)
TVICS19       CSS ,                       -- NOT USED --
*
TVI950        TERM TVII,FLAGS=X'0C00',DELAY=100,SGR=ADM31MOD
              CSS  EQU=TVICS1
              CSS  EQU=TVICS2
              CSS  EQU=TVICS3
              CSS  EQU=TVICS4
              CSS  EQU=TVICS5
              CSS  EQU=TVICS6
              CSS  EQU=TVICS7
              CSS  EQU=TVICS8
              CSS  EQU=TVICS9
              CSS  EQU=TVICS10
              CSS  EQU=TVICS11
              CSS  EQU=TVICS12
              CSS  EQU=TVICS13
              CSS  EQU=TVICS14
              CSS  EQU=TVICS15
              CSS  EQU=TVICS16
              CSS  EQU=TVICS17
              CSS  EQU=TVICS18B
              CSS  EQU=TVICS19
TVI950R       TERM TVII,FLAGS=X'0800',DELAY=100,SGR=ADM31ATR
              CSS  EQU=TVICS1
              CSS  EQU=TVICS2
              CSS  EQU=TVICS3
              CSS  EQU=TVICS4
              CSS  EQU=TVICS5
              CSS  EQU=TVICS6
              CSS  EQU=TVICS7
              CSS  EQU=TVICS8
              CSS  EQU=TVICS9
              CSS  EQU=TVICS10
              CSS  EQU=TVICS11
              CSS  EQU=TVICS12
              CSS  EQU=TVICS13
              CSS  EQU=TVICS14
              CSS  EQU=TVICS15
              CSS  EQU=TVICS16
```

```
         CSS   EQU=TVICS17
         CSS   EQU=TVICS18A
         CSS   EQU=TVICS19
*
TVII     NODE SOH,GOTO,TVISOH "FUNCT" KEY INTRODUCER
         NODE ESC,GOTO,TVIESC ESCAPE SEQUENCE
         NODE SUB,CALL,CLEAR  "CLEAR"
         NODE RS,CALL,HOME    "HOME"
         NODE LF,CALL,NL      "LINEFEED"
         NODE SYN,CALL,DOWN
         NODE OTHERS,GOTO,ADMNORM      REST LIKE ADM31
         NODE SEQENDER,GOTO,*
*    ******************************************************
*    * FUNCT SEQUENCE STATE  (PFKS)                       *
*    ******************************************************
TVISOH   NODE INDEX,GOTO,TVIPFK1,PFK1
         NODE NULL,GOTO,FUNCA
         NODE INDEX,GOTO,TVIPFK2,PFK13
         NODE NULL,GOTO,FUNCA
         NODE COMMA,GOTO,FUNCA,PA1
         NODE LESS,GOTO,FUNCA,PA1
         NODE PERIOD,GOTO,FUNCA,PA2
         NODE GREATER,GOTO,FUNCA,PA2
         NODE SLASH,GOTO,FUNCA,PA3
         NODE QUESTION,GOTO,FUNCA,PA3
         NODE SEQENDER,GOTO,*
*    ******************************************************
*    * ESCAPE SEQUENCE TABLE                              *
*    ******************************************************
TVIESC   NODE ESC,GOTO,A31EESC
         NODE O,GOTO,TVISET   "LINE DELETE" (SHIFT)
         NODE Q,CALL,PFK12    "CHAR INSERT"
         NODE LCQ,CALL,PFK24  "CHAR INSERT" (SHIFT)
         NODE W,CALL,NULL     "CHAR DELETE"
         NODE E,CALL,INSRT    "LINE INSERT"
         NODE R,CALL,EEOF     "LINE DELETE"
         NODE T,CALL,TAB      "LINE ERASE"
         NODE LCT,CALL,BTAB   "LINE ERASE"  (SHIFT)
         NODE Y,CALL,INDENT   "PAGE ERASE"
         NODE LCY,CALL,UNDENT "PAGE ERASE"  (SHIFT)
         NODE @7,CALL,CURSEL  "SEND PAGE"
         NODE @6,CALL,TREQ    "SEND PAGE"   (SHIFT)
         NODE I,CALL,COLBAK   "BACKTAB"
         NODE ASTERISK,CALL,RESHOW
         NODE SEQENDER,GOTO,*
*    ******************************************************
*    * FUNCTION SEQUENCE TERMINATOR FOR SETUP INTRODUCER  *
*    ******************************************************
TVISET   NODE RS,CALL,SETHOM
         NODE LF,CALL,SETMRG
         NODE SUB,CALL,CLRTAB
         NODE OTHERS,GOTO,COMSET
TVIPFK1  CSS  '@ABCDEFGHIJ'
TVIPFK2  CSS  ACCENT,'abcdefghij'
*
```

```
************************************************************ *
* TYPEWRITER TERMINAL TABLE                                  *
* AN ATTEMPT TO MAKE THE AVERAGE ASCII TYPEWRITER TERMINAL   *
* DEVICE USEFUL                                              *
************************************************************ *
TYPETERM     TERM TPTMI,FLAGS=X'8000'
             CSS (HARDCOPY)              POSITION "CURSOR"
             CSS ,                       ERASE EOL
             CSS ,                       LOCAL PRINT
             CSS BEL                     TONE
             CSS BS                      CURSOR LEFT
             CSS BLANK                   CURSOR RIGHT
             CSS ,                       -- NOT USED --
             CSS ,                       -- NOT USED --
             CSS ,                       SIGNAL INSERT MODE
             CSS ,                       SIGNAL END INSERT MODE
             CSS ,                       DISCONNECT
             CSS CR,LF,LF,LF             CLEAR
             CSS CR,FF                   "TERMINAL" INITIALIZE
             CSS COLON                   ILLEGAL ASCII CHAR
             CSS LCN,BS,LCZ              ILLEGAL APL CHAR
             CSS ,                       APL CHARS ON
             CSS ,                       APL CHARS OFF
             CSS ,                       DISPLAY MODE
             CSS ,                       -- NOT USED --
*       *****************************************************
*       * INPUT PARSE TABLE                                 *
*       *****************************************************
TPTMI        NODE ESC,GOTO,TPTMESC       ESCAPE SEQUENCE
             NODE LF,CALL,NL             "NEW LINE" KEY
             NODE BS,CALL,LEFT
             NODE DEL,CALL,DELETE
             NODE HT,CALL,COLTAB         COLUMN TAB
             NODE CR,CALL,ENTER
             NODE SYN,CALL,RESHOW        CTRL-V
             NODE SEQENDER,GOTO,*
*       *****************************************************
*       * ESCAPE SEQUENCE TABLE                             *
*       *****************************************************
TPTMESC      NODE I,CALL,INSRT           INSERT TOGGLE KEY
             NODE LCI,CALL,INSRT         INSERT TOGGLE KEY
             NODE B,CALL,BTAB            BACKTAB
             NODE LCB,CALL,BTAB          BACKTAB
             NODE T,CALL,TAB             TAB
             NODE LCT,CALL,TAB           TAB
             NODE HT,CALL,COLBAK         COLUMN BACKTAB
             NODE OTHERS,GOTO,TYPPFK
*
```

```
************************************************************
*                     RSCS LINEDRIVER HARDCOPY STATEMENTS
******************** DO NOT CHANGE THIS SECTION ! ************
*
*********************************************************** *
* HARDCOPY DEVICE TABLE                                     *
* - REQUIRED WHEN USING RSCS DRIVEN PRINTER/PLOTTER DEVICES - *
*********************************************************** *
HARDCOPY     TERM HCPYI,FLAGS=X'8000'
             CSS (HARDCOPY)
             CSS ,
             CSS ,
             CSS BEL
             CSS BS
             CSS BLANK
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS CR,LF,LF,LF
             CSS CR,FF
             CSS COLON
             CSS LCN,BS,LCZ
             CSS ,
             CSS ,
             CSS ,
             CSS ,
*       ***************************************************
*       * INPUT PARSE TABLE                                 *
*       ***************************************************
HCPYI        NODE ESC,GOTO,HCPYESC      ESCAPE SEQUENCE
             NODE LF,CALL,NL            "NEW LINE" KEY
             NODE BS,CALL,LEFT
             NODE DEL,CALL,DELETE
             NODE HT,CALL,COLTAB        COLUMN TAB
             NODE CR,CALL,ENTER
             NODE SYN,CALL,RESHOW       CTRL-V
             NODE SEQENDER,GOTO,*
*       ***************************************************
*       * ESCAPE SEQUENCE TABLE                             *
*       ***************************************************
HCPYESC      NODE I,CALL,INSRT          INSERT TOGGLE KEY
             NODE LCI,CALL,INSRT        INSERT TOGGLE KEY
             NODE B,CALL,BTAB           BACKTAB
             NODE LCB,CALL,BTAB         BACKTAB
             NODE T,CALL,TAB            TAB
             NODE LCT,CALL,TAB          TAB
             NODE HT,CALL,COLBAK        COLUMN BACKTAB
             NODE OTHERS,GOTO,TYPPFK
```

```
*********************************************************** *
*   PLOTTER - HARDCOPY DEVICE TABLE                         *
* - REQUIRED WHEN USING RSCS DRIVEN PRINTER/PLOTTER DEVICES - *
*********************************************************** *
PLOTTER      TERM HCPYI,FLAGS=X'8000'
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
             CSS ,
```

```
                 TERMINAL TRANSLATE TABLES 7E MOD
TTRAN7E  DS    0A
***************************************************************
* THE FOLLOWING TABLES TRANSLATE EACH CHARACTER               *
* MOVED FROM THE RATS TO THE OUTPUT BUFFER (FOR STATIC MODES) *
* THEY TAKE EXTENDED ASCII INPUT AND YIELD ASCII OUTPUT       *
* SINCE ASCII IS SEVEN BIT CODE, THE HIGH ORDER BIT ON        *
* INDICATES SPECIAL HANDLING. THE SPECIAL HANDLING CODES      *
* ARE NEGATIVE INDEXES INTO A BRANCH TABLE BASED ON THE       *
* FOLLOWING ASSIGNMENTS                                       *
*     VALUE   MEANING                                         *
*      FF     CHARACTER IS ONLY IN OTHER (APL/NONAPL) SET     *
*      FE     CHARACTER IS UNDERSCORED ALPHABETIC             *
*      FD     CHARACTER IS GENERAL OVERSTRIKE                 *
*      FC     CHARACTER IS ILLEGAL GRAPHIC                    *
*      FB     CHARACTER IS ATTRIBUTE BYTE                     *
*      FA     CHARACTER IS LOWERCASE (APL TABLES ONLY)        *
*      F9     CHARACTER IS PRINTER GRAPHIC                    *
***************************************************************

***************************************************************
*        NORMAL (NON-APL) ASCII TABLE                         *
***************************************************************
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'20FCFCFCFCFCFCFCFCFCFCFCFCFCFCFC' 0
         DC    X'FCFCFCFCFCFCFCFCFCFCFCFCFCFCFCFC' 1
         DC    X'202122232425262728292A2B2C2D2E2F' 2
         DC    X'303132333435363738393A3B3C3D3E3F' 3
         DC    X'404142434445464748494A4B4C4D4E4F' 4
         DC    X'505152535455565758595A5B5C5D5E5F' 5
         DC    X'606162636465666768696A6B6C6D6E6F' 6
         DC    X'707172737475767778797A7B7C7D7E3B' 7
         DC    X'2AF9F9F9F95C7C7B7DFFFFFCFCFFFFFF' 8
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF' 9
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF' A
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF' B
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF' C
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFC' D
         DC    X'FBFBFBFBFBFBFBFBFBFBFBFBFBFBFBFB' E
         DC    X'FBFBFBFBFBFBFBFBFBFBFBFBFBFBFBFB' F
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F

***************************************************************
* NORMAL ASCII INPUT TRANSLATE TABLE                          *
* NOTE: X'00' INDICATES CONTROL STRING INTRODUCER             *
***************************************************************
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'00000000000000000000000000000000' 0
         DC    X'00000000000000000000000000000000' 1
         DC    X'202122232425262728292A2B2C2D2E2F' 2
         DC    X'303132333435363738393A3B3C3D3E3F' 3
         DC    X'404142434445464748494A4B4C4D4E4F' 4
         DC    X'505152535455565758595A5B5C5D5E5F' 5
         DC    X'606162636465666768696A6B6C6D6E6F' 6
         DC    X'707172737475767778797A7B7C7D0000' 7
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F

***************************************************************
*        TYPEWRITER PAIRED APL TABLE                          *
***************************************************************
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'20FCFCFCFCFCFCFCFCFCFCFCFCFCFCFC' 0
         DC    X'FCFCFCFCFCFCFCFCFCFCFCFCFCFCFCFC' 1
         DC    X'20FFFFFF7EFFFF4B3A22502D2C5F2E2F' 2
         DC    X'303132333435363738393E3C23252651' 3
```

```
         DC    X'FF6162636465666768696A6B6C6D6E6F' 4
         DC    X'707172737475767778797AFF3FFFFF46' 5
         DC    X'FFFAFAFAFAFAFAFAFAFAFAFAFAFAFAFA' 6
         DC    X'FAFAFAFAFAFAFAFAFAFAFAFA7B4D7DFF3C' 7
         DC    X'50F9F9F9F93F4D7B7D3B27FCFCFDFDFD' 8
         DC    X'FDFDFDFDFDFDFDFDFDFDFDFDFDFDFDFD' 9
         DC    X'404142434445214748494A284C4D4E4F' A
         DC    X'292A52535455565758595A5B5C5D5E2B' B
         DC    X'60FEFEFEFEFEFEFEFEFEFEFEFEFEFEFE' C
         DC    X'FEFEFEFEFEFEFEFEFEFEFEFE3D7C3F24FC' D
         DC    X'FBFBFBFBFBFBFBFBFBFBFBFBFBFBFBFB' E
         DC    X'FBFBFBFBFBFBFBFBFBFBFBFBFBFBFBFB' F
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F

**************************************************************
* TYPEWRITER PAIRED APL INPUT TRANSLATE TABLE                *
**************************************************************
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'00000000000000000000000000000000' 0
         DC    X'00000000000000000000000000000000' 1
         DC    X'20A6293CDE3D3E8AABB0B1BF2C2B2E2F' 2
         DC    X'303132333435363738392889 3BDB3ADD' 3
         DC    X'A0A1A2A3A4A55FA7A8A9AA27ACADAEAF' 4
         DC    X'2A3FB2B3B4B5B6B7B8B9BABBBCBDBE2D' 5
         DC    X'C0414243444546474849 4A4B4C4D4E4F' 6
         DC    X'505152535455565758595A7BDC7D2400' 7
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F

**************************************************************
*        BIT PAIRED APL TABLE                                *
**************************************************************
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'20FCFCFCFCFCFCFCFCFCFCFCFCFCFCFC' 0
         DC    X'FCFCFCFCFCFCFCFCFCFCFCFCFCFCFCFC' 1
         DC    X'20FFFFFF7CFFFF4B2B2A502D2C3D2E2F' 2
         DC    X'303132333435363738393E3C23252751' 3
         DC    X'FF6162636465666768696A6B6C6D6E6F' 4
         DC    X'707172737475767778797AFF3FFFFF46' 5
         DC    X'FFFAFAFAFAFAFAFAFAFAFAFAFAFAFAFA' 6
         DC    X'FAFAFAFAFAFAFAFAFAFAFAFA5D4D7DFF3C' 7
         DC    X'50F9F9F9F93F4D5D7D3B3AFCFCFDFDFD' 8
         DC    X'FDFDFDFDFDFDFDFDFDFDFDFDFDFDFDFD' 9
         DC    X'224142434445214748494A294C4D4E4F' A
         DC    X'5F2852535455565758595A405B60267E' B
         DC    X'5CFEFEFEFEFEFEFEFEFEFEFEFEFEFEFE' C
         DC    X'FEFEFEFEFEFEFEFEFEFEFEFE5E7B3F24FC' D
         DC    X'FBFBFBFBFBFBFBFBFBFBFBFBFBFBFBFB' E
         DC    X'FBFBFBFBFBFBFBFBFBFBFBFBFBFBFBFB' F
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F

**************************************************************
* BIT PAIRED APL INPUT TRANSLATE TABLE                       *
**************************************************************
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F
         DC    X'00000000000000000000000000000000' 0
         DC    X'00000000000000000000000000000000' 1
         DC    X'20A6A03CDE3DBE3EB1AB29282C2B2E2F' 2
         DC    X'303132333435363738398A893B2D3ADD' 3
         DC    X'BBA1A2A3A4A55FA7A8A9AA27ACADAEAF' 4
         DC    X'2A3FB2B3B4B5B6B7B8B9BABCC07BDBB0' 5
         DC    X'BD4142434445464748494A4B4C4D4E4F' 6
         DC    X'505152535455565758595ADC247DBF00' 7
*                 0 1 2 3 4 5 6 7 8 9 A B C D E F
         END
```

Index

E

F

G

H

I

K

L

M

N

O

P

R

S

T

IBM 7171 ASCII DACU Reference Manual and Programming Guide Printed in U.S.A. GA24-4020-0

GA24-4020-0

IBM 7171 ASCII DACU
Reference Manual and Programming Guide
Order No. GA24-4020-0

READER'S COMMENT FORM

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate. Comments may be written in your own language; English is not required.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

	Yes	*No*
• Does the publication meet your needs?	☐	☐
• Did you find the material:		
Easy to read and understand?	☐	☐
Organized for convenient use?	☐	☐
Complete?	☐	☐
Well illustrated?	☐	☐
Written for your technical level?	☐	☐

• What is your occupation? ______________________________

• How do you use this publication:

As an introduction to the subject?	☐	As an instructor in class?	☐
For advanced knowledge of the subject?	☐	As a student in class?	☐
To learn about operating procedures?	☐	As a reference manual?	☐

Your comments:

If you would like a reply, please supply your name and address on the reverse side of this form.

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

GA24-4020-0

Reader's Comment Form

Fold and Tape — Please Do Not Staple — Fold and Tape

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department X00
P. O. Box 6
Endicott, New York 13760

Fold — Fold

If you would like a reply, *please print:*

Your Name ______________________
Company Name ______________________ *Department* ______________________
Street Address ______________________
City ______________________
State ______________________ *Zip Code* ______________________
IBM Branch Office serving you ______________________

IBM®

Cut or Fold Along Line

GA24-4020-0

IBM 7171 ASCII DACU Reference Manual and Programming Guide Printed in U.S.A. GA24-4020-0

GA24-4020-00

IBM®