Data General

DASHER D3 DISPLAY TERMINAL
PROGRAMMER'S REFERENCE

*** preliminary**

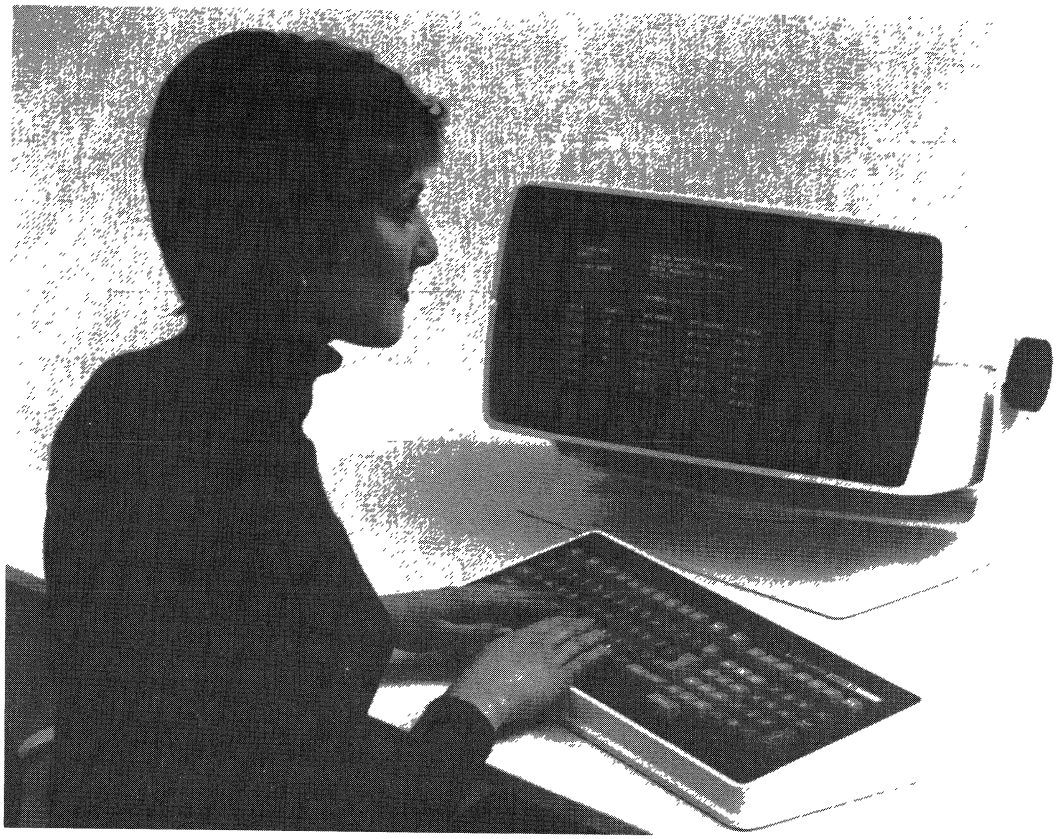
055-203-00

Data General Corporation, Westboro, Massachusetts 01581

Printed in U.S.A.

* * *

TABLE OF CONTENTS



A Dasher D3 (6093) Display Terminal in operation

The Dasher D3 (6093) Display Terminal preserves the basic layout of previous Dasher terminals with its tilting, swivelling screen, and separate keyboard.

The D3 keyboard incorporates significant improvements. Its keys are sculptured for better operator "feel". There are more user-defined function keys (18 keys for 72 functions). Dedicated keys for ENTER, NEGATIVE ENTER and minus sign have been added next to the numeric pad. Legends for typical word-processing editing functions have been screened on the key fronts. These editing functions are implemented witnin the terminal in buffered mode.

Yet the keyboard is still completely compatible with all Dasher D2 (6053) functions. Every keyboard function has been preserved on the newer D3, though some keys have been relocated for more convenient operator usage.

INTRODUCTION

The Dasher D3 is a microprocessor-based, stored-program control, cathode ray display terminal. It has been designed as a functional superset of the Dasher D2 Terminal. The results of these additional functions mean simplified operating procedures, improved data entry characteristics, and greater text editing capability than previously available with the Dasher D2.

The Dasher D3 allows a broad range of operating parameters to be controlled by the terminal. Moreover these operating parameters, which are under program control, can easily be altered, providing the flexibility needed for demanding, on-line applications.

The 18 user defined function keys provide an enormous degree of flexiblity (up to 72 functions) in application programming.

Architectually, the Dasher D3 was designed for ease of use and operator convenience. A detached, sculptured, typewriter-style keyboard helps operators or data entry personnel to easily locate often used keys. The keyboard has been organized for minimum movement in locating keys and variation of key characteristics (color, size, shape, and height) make searching simple. The extended numeric keyboard parallels a standard calculator layout, further simplifying use.

Screen management with the Dasher D3 is easily accomplished with a number of powerful features. For improved form design, layout, and text manipulation, these features include programmable character blink, underscore, dual-intensity, block fill, reverse video, and character/screen protection.

The Dasher D3's "character pacing" assures control over processor interrupts keeping the host processor overhead to a minimum.

The Dasher D3's "pass-through" feature provides the ability of controlling a serial printer from the host computer without interfering with the Dashers' screen contents. The buffered mode provides great flexibility and improved system performance for the system builder who utilizes its functions. Several keys are already labeled with typical text manipulation functions should the system builder decide to implement these functions.

KEYBOARD FUNCTIONS

The Dasher D3 (6093) Display Terminal provides the same key entry functions for interactive mode as the Dasher D2 (6053) model and in addition has powerful buffered mode features. Figure 1 shows the basic compatability between the two models. Note that some of the keys have been repositioned on the D3 model for improved operator convenience.

The buffered mode is designed for the entry and editing of data by fields. These fields can be of two types: protected and unprotected.

A protected field is a portion of screen area that has all its data protected against operator entry (see discussion of Protected Data attribute under heading: Character Attributes).

An unprotected field can be either a full screen line of unprotected data or a portion of a line between a protected field and another protected field. The beginning or ending of a line can also constitute field boundaries.

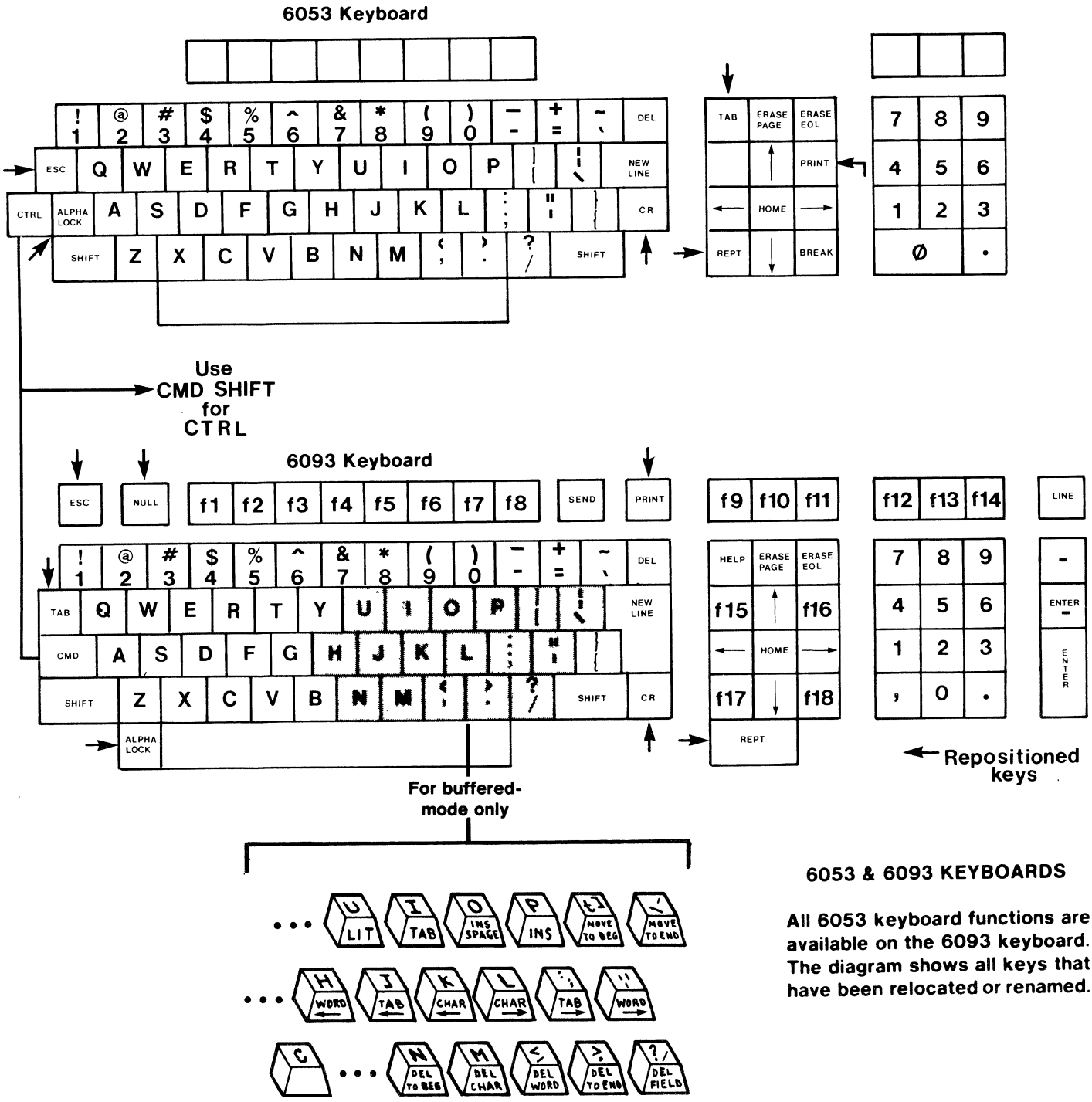
The buffered mode has a set of powerful screen editing functions such as delete and insert. These functions are accessed using the CMD key in conjunction with centrally-located data keys (see Figure 1). These functions are described below.

MOVE CURSOR RIGHT ONE CHARACTER
Keying - CHAR --> (CMD L)
Function - Moves the cursor right one position; if already at end of line or unprotected field, then the bell sounds. Cursor cannot enter beyond first location of unfilled area.

MOVE CURSOR RIGHT TO NEXT TAB STOP
Keying - TAB --> (CMD ;)
Function -Moves the cursor to next tab stop or end of entered data whichever comes first. If cursor is at end of entered data in the field, then the bell sounds and no action occurs.

MOVE CURSOR RIGHT TO NEXT WORD
Keying - WORD --> (CMD ')
Function - Moves cursor to beginning of next word or to the end of entered data in the field. If at end of entered data, field, or line, the bell sounds and no action occurs.

MOVE CURSOR TO END OF FIELD
Keying - MOVE TO END (CMD \)
Function - Moves cursor to last position in the field or to the first location of unfilled area in the field.



6053 & 6093 KEYBOARDS

All 6053 keyboard functions are available on the 6093 keyboard. The diagram shows all keys that have been relocated or renamed.

Figure 1 Dasher D3 (6093) Display Terminal Keyboard

MOVE CURSOR LEFT ONE CHARACTER
Keying - CHAR <-- (CMD K)
Function - Moves cursor one position to the left. If cursor is at beginning of the field, the bell sounds and cursor position remains unchanged.

MOVE CURSOR TO PREVIOUS TAB STOP
Keying - TAB <-- (CMD J)
Function - Moves cursor to previous tab stop or to beginning of the field. If cursor is at beginning of a field, the bell sounds.

MOVE CURSOR TO BEGINNING OF PREVIOUS WORD
Keying - WORD <-- (CMD H)
Function - If The cursor is in the middle of a word or between words, it moves to beginning of the current word. If the cursor is a the beginning of a word, it moves to the beginning of the previous word or field. If curssor is at the beginning of a field, the bell sounds.

MOVE CURSOR TO BEGINNING OF FIELD
Keying - MOVE TO BEG (CMD [)
Function - Moves cursor to beginning of the current field.

MOVE CURSOR TO NEXT FIELD
Keying - ---> (on Cursor Control Pad) or CMD X
Function - Moves cursor to first location in next unprotected field. If there are no protected fields, the cursor moves to beginning of the next line. If in last field of screen, cursor moves to the first unprotected field at top of screen.

MOVE CURSOR TO PREVIOUS FIELD
Keying - <-- (on Cursor Control Pad) or CMD Y
Function - Moves cursor to first character location of previous unprotected field.

COMPLEMENT INSERT MODE
Keying - INS (CMD P)
Function- Alternate action, either enables or disables Insert mode with each keying. Insert mode is disabled at entry to buffered mode or whenever a new field is entered. When in Insert mode, as each key is struck (except for control keys), the characters starting at and to right of the cursor shift one position to the right. The character struck is inserted at the cursor location and the cursor moves one position to the right. Any character in the last position in the line or field is lost when the shift occurs. When the insert mode is disabled, new data entry writes over and destroys the existing data.

INSERT TAB
Keying - TAB (CMD I) or normal TAB key
Function- Characters at and to right of the cursor are shifted right to the next tab stop and the resulting space is filled with space codes.These spaces are flagged and are replaced by a single tab character before the data is transmitted to the host. However, if a Replace Tab instruction (from host) is stored in the Tab Read Mode Register, then the space codes are retained and transmitted when the Read Screen instruction occurs. The bell also sounds whenever a full field condition exists and additional Insert Tab functions are keyed. The cursor stays at end of tab field.

INSERT SPACE
Keying - INS SPACE (CMD O)
Function - Characters to the right of the cursor are shifted right one position and a space is inserted. If the field is full, the bell sounds and no action occurs.

DELETE CURRENT CHARACTER
Keying - DEL CHAR (CMD M)
Function - Deletes the character at cursor location and shifts subsequent characters left one position. The last position in the field is filled with the fill character. If the cursor is at end of the field, the last character is deleted and subsequent deletions will sound the bell and no other action occurs.

DELETE PREVIOUS CHARACTER
Keying - DEL
Function - The character previous to the cursor location is replaced with a space code. If at end of entered data, the current fill character is the replacement character. If the cursor is at last character in a full field, then the character at the cursor location is replaced with a fill character. If the cursor is at the beginning of the field, the bell sounds and no action occurs.

DELETE NEXT WORD
Keying - DEL WORD (CMD ,)
Function - Deletes characters from cursor location to beginning of next word. Subsequent characters in the field shift left to closeup space and fill characters are inserted at the end to complete the field. If cursor is at end of the field, the bell sounds and no action occurs.

DELETE TO BEGINNING OF FIELD
Keying - DEL TO BEG (CMD N)
Function - Deletes characters between cursor and beginning of field. Subsequent characters in field and cursor are shifted left to beginning of field. Fill characters are inserted at end to complete field. If cursor is at beginning of the field, the bell sounds and no action occurs.

DELETE TO END OF FIELD

Keying - DEL TO END (CMD .)

Function - Inserts fill characters from cursor location to end of field. If cursor is at end of field or at end of entered data, the bell sounds and no action occurs.

DELETE ENTIRE FIELD

Keying - DEL FIELD (CMD /)

Function - Entire field is filled with fill characters and cursor is repositioned to beginning of field.

DELETE PREVIOUS WORD

Keying - WORD (CMD DEL)

Function- If cursor is pointing to a word separator (space code or any punctuation code) or to the first character of a word, the previous word is deleted; otherwise the same word is deleted. Subsequent data along with cursor are shifted left to close up space. Fill characters are inserted at end to fill the vacated locations. If cursor is at beginning of field, the bell sounds and no action occurs.

PROGRAMMING

INTRODUCTION

This section provides the system programmer with reference information to be used in preparing an interface program for the Dasher D3 Display Terminal. The section starts out with basic information, covering functional aspects of the terminal which are of concern to the programmer. Subsequent section areas describe the instructions that the host can use to control terminal operations.

For condensed code and instruction listings, refer to the Appendices in the rear.

OVERVIEW OF TERMINAL OPERATIONS

The terminal functions as a combined data entry and display device for an associated host computer system. In this, it receives control and conditioning from the host system and responds to keyboard entry by generating appropriate ASCII code. How this code is displayed is dependent on which of two basic operating modes is being used. The two modes are: interactive mode and buffered mode.

In interactive mode, the codes are individually transferred to the host as each key is pressed. In turn, the host system echoes each data code back to the terminal for display.

In buffered mode, as each code is generated it is stored in the terminal and immediately displayed by the terminal for operator viewing. No host action is required. After the data has been corrected and edited, the operator can transfer control to the host. Generally, the host will read the accumulated data.

From the viewpoint of the host system, certain internal functions of the terminal are of concern to the programmer. Basically, these concerns relate to control of the terminal itself and also of the display of data on the screen. The following paragraphs discuss some of these concerns.

Control Registers and Flags

Several registers internal to the terminal are associated with functions controllable by the host. Among these registers are six status registers which can be written to or read out of by the host. The status registers are: Terminal Mode, Keyboard Lock, Tab Read Mode, Tab Interval, Fill Character, and Terminal Number.

Three other registers also available to the host are: the Delimiter Table, the Current Attribute Register, and the Cursor Address Register.

In addition, three status flags enable character blinking, Roll Mode (scrolling of the screen), and Insert Mode. Basic functions of the various registers and flags are as follows:

*Terminal Mode Register - Identifies the operating mode of the terminal. The codes are: 000 for Interactive Mode; 177 for buffered mode.

*Keyboard Lock Register - Identifies whether keyboard entry is permitted. The code 000 = keyboard unlocked; 177 = keyboard locked.

*Tab Read Mode Register - Indicates whether tab codes will be sent to host on a screen read operation or if the codes will be replaced with an appropriate number of space codes. The codes are: 001 for replace tab codes; 177 for send tab codes.

*Tab Interval Register - Stores the octal value (number of character positions) between tab stops.

*Fill Character Register - Stores the ASCII code for the fill character.

* Terminal Number Register - Stores the assigned number (octal) of the terminal.

*Delimiter Table - Defines up to 128 delimiters which can be used by the operator in buffered mode operation to transfer control to the host.

*Current Attribute Register - Stores seven bits which define the screen attributes for current data entry. Function of the individual attributes is described under the heading: Character Attributes.

*Cursor Address Register - Stores the current screen address of the cursor. Any change in cursor location is obtained by writing a new address into the register.

*Blink Enable Flag - When set, enables any character with the blink attribute bit set to blink. Conversely, when cleared, all blinking (except for cursor) is disabled.

*Roll Enable Flag - When set, enables the screen to roll (scroll) a line each time the cursor is moved beyond the 24th line.When the flag is cleared, roll mode is disabled and page mode is enabled. In the page mode, when the last screen location is exceeded, the cursor moves to the top of the screen.

Display Concept

The screen display can consist of up to 24 lines of 80 characters each. In buffered mode the data is immediately displayed as it is keyed. In interactive mode, the host must echo the keyed data codes back to the terminal for display.

Cursor Addressing

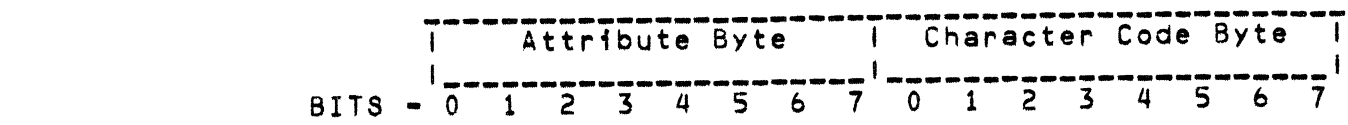
As each data code is received, it is displayed at the current cursor address and then the cursor is advanced to the next screen location. The cursor address is stored in the two byte Cursor Address Register.

In interactive mode, the cursor can be relocated to any addressable screen location by a command from the host. The screen address is defined by a three byte sequence; the first byte, 037, identifies that a cursor address follows. The second byte defines the column address (horizontal character position) on the screen and can consist of any number from 0 to 79 (octal 000 to 117). The third byte defines the row address (vertical line position) on the screen and can consist of any number from 0 to 23 (octal 000 to 027). If desired, the host can read the cursor address and from that calculate the repositioning.

In buffered mode, the cursor location is controlled directly from the keyboard.

Character Attributes

The individual characters on the display can have various attributes, such as blink, dim, underscore, etc. As each character is received its attributes are defined according to the contents of the Current Attribute Register. To accomodate this, the display data is stored in the terminal in a two-byte format, with one byte containing the character code and the other byte containing the attribute information. The bit arrangement as it appears at the terminal interface is as follows:



Attribute Byte Bit Assignment:		Function:
Bit 7	block fill	7x10 character space is illuminated
Bit 6	blink	displayed character blinks on and off
Bit 5	reverse video	displayed character is black on white
Bit 4	dim	displayed character has reduced intensity
Bit 3	underscore	displayed character is underscored
Bit 2	protected data	indicates character cannot be changed from keyboard.
Bit 1	modified data	indicates character code has changed. It is set by terminal when any data in field has changed.
Bit 0	tab stop	indicates a tab stop location (not a character attribute) which is set by the terminal according to the contents of the Tab Interval Register. Not accessable by the host.

Character Byte Bit Assignment:

- Bits 1 thru 7 - character code
- Bit 0 - signifies a tab space which is sent to the host on a screen read operation.

Attribute bits 1 thru 7 are controllable from the host. Bit 0 of both the attribute byte and the character byte are controlled from within the terminal. In operation, a current attribute register in the terminal stores the current host-defined attributes. These are imparted to the new data as it is entered from the keyboard. In addition, the modified data attribute is set by the terminal (see description below). Also see the Field Control instructions for additional information on attribute application.

In the set attribute commands, a combination of masking and XOR functions are used to achieve the desired bit configurations. Each bit of the current attributes is ANDed with the corresponding bit of the mask byte and then exclusive ORed with the XOR byte to produce the new current attributes. The results can be predicted as follows:

Mask Bit	XOR Bit	Resulting Attribute Bit
0	0	0 (cleared)
0	1	1 (set on)
1	0	no change
1	1	complement

In addition to the capability of defining all the attributes at once with a direct write to the Current Attribute Register, they can be individually defined by specific commands from the host. Also, a Set Attribute String command can redefine the attributes of a specified screen area. Other commands provide for reading either current attributes or those of a specified area on the screen.

The various attributes are individually described in the following paragraphs.

Character Blinking - Blinking of displayed characters is defined on an individual character basis by bit 6 of the character's attribute byte. Whether the characters actually blink is determined by the Blink Enable Flag. Blinking is enabled when the flag is set; conversely, with the flag cleared, blinking is prohibited.

In addition to the appropriate blink commands, each time an ERASE PAGE command is generated the Blink Enable Flag is set and the display is blink enabled.

Character Dimming/Underscoring - Bits 3 and 4 of the individual character's attribute byte control underscoring and dimming, respectively. Thus, each character whose underscore bit is set will be underscored on the display, Similarly, if the dim bit is set, the character will be display with reduced intensity.

An ERASE PAGE instruction clears both the dim and underscore bits in the Current Attribute Register.

Block Fill - The Block Fill attribute is controlled by bit 7 of the character's attribute byte. When the bit is set, the complete 7x10 dot character field is illuminated with the result that the displayed image of the associated character is effectively masked out.

Reverse Video - The Reverse Video attribute is controlled by bit 5 of the character's attribute byte. When the bit is set, the character image is outlined in black on a 5x7 dot illuminated background.

Protected Data - The Protected Data attribute is controlled by bit 2 of the character's attribute byte. When the bit is set, the character is protected from being overwritten or moved from its existing screen location. Several characters in sequence with the Protected Data attribute set constitute a protected field. In contrast, data locations between protected fields are called unprotected fields.

Modified Data - The Modified Data attribute (bit 1 of the character's attribute byte) is set for all characters in the field whenever any change is made to existing data in the field. This is essentially a flag to enable selective reading of screen data by the host. The Modified Data attribute is set by the terminal and can only be cleared by an instruction from the host.

Tab Stop/Read Functions

Tabs are treated in one of two ways, depending upon the host-set value of the Tab Read Mode Register. If the register is set to 001, space codes that fill from the end of data to the next tab stop are identified within the terminal by bit 0 of the character code byte. On a read screen operation they are replaced as a group by a single tab code.

If the register is set to 177, tabbing results in the storage of ASCII space codes. On a read screen, these space codes are transmitted rather than the single tab code.

Tab Stop locations on the screen are flagged inside the terminal by the setting of bit 0 on the individual character attribute bytes. The interval between stops is defined by the Tab Interval Register. The register is initialized with the Sysgen value, but can be changed by an instruction from the host.

Roll Mode (Scrolling)

A Roll Enable flag in the terminal defines whether the screen will be in Roll Mode (permits scrolling) or Page Mode (scrolling prohibited).This flag is set or cleared by host-supplied instructions. When the display is roll enabled, the screen rolls up one line each time the cursor overflows the bottom line or a New Line code is received when the cursor is on the bottom line. When the display is not roll enabled, the cursor moves to

the first character position on the top line each time the cursor overflows the bottom line or a New Line code is received when the cursor is on the bottom line.

The ERASE PAGE command does not affect the Roll Enable flag.

Field Control

In some data entry applications, a prepared form is sent to the screen by the host system. Those areas of the form where the operator is expected to enter data are called unprotected fields. These fields can be identifed to the operator by fill characters which are overwritten as new data is entered.

Fixed areas of the form which the operator cannot change are called protected fields and are identified within the terminal by the protect attribute bit. Other bits flag the block fill, blink, reverse video, dim, underscore, and modified data attributes.

The modified data attribute can be used to select for transfer to the host system only that data which has undergone change. After such a reading by the host, the host would most likely want to reset the modified attribute bits on data which was read so that new or modified data can be discriminated.

Since, in buffered mode, data entry is processed internally in the terminal, the nost must be informed when data is ready for transfer to the host. For this purpose, the host system can establish delimiters codes which when generated at the terminal will inform tne host that data should be read. In this manner, the host can be informed when a field has been completed, or perhaps a particular grouping of fields, or of the full screen.

The commands that the host can use to define and control fields within the screen are described under the heading: Screen Control. Also refer to the information under headings: Character Attributes, Attribute Control.

CODING CONVENTIONS

Instruction format to terminal: 036 <instruction code> plus any required data or positioning bytes. May or may not have a terminator.

Terminal response format: 036 176 <function> plus any required data or cursor coordinates and terminator.

Data entry format, keyboard to host: 8-bit ASCII code consisting of 7 data bits and 1 parity bit. Refer to the ASCII code listing in the Appendix.

Function code format, keyboard to host: 036 <function code> The various user function codes available to the programmer are listed in Appendix C.

Cursor position format: 037 <X> <Y> where: X = column position
Y = row position
Minimums: X = 0
Y = 0
Maximums: X = 79 (117 octal)
Y = 23 (27 Octal)

HOST/TERMINAL INSTRUCTIONS

The following instructions can be used by the host to control terminal operations. For a conceptual understanding of the various functions, refer to the description given under heading: OVERVIEW OF TERMINAL OPERATIONS. Note that instructions which are compatible with the Data General 6053 Model Terminal are so indicated.

Refer to the Appendices for ASCII code listings, key layout and code assignments (including user function keys), extended graphic character symbols and codes, and Sysgen information.

Terminal Management

MASTER RESET
- Code to terminal - 036 025 (terminal reset from host)
- Keying - CMD LINE (terminal reset from keyboard)
- Function - Reinitializes registers and flags in the terminal and loads Sysgen parameters. Unlocks keyboard.

ID REQUEST
- Code to terminal - 036 012
- Function - Requests terminal for following identification data: terminal type (040=model 6093), revision level of terminal, and assigned terminal number.
- Terminal Response - 036 176 043 040 <Rev.#><term.#> 036 176 057

UNLOCK KEYBOARD
- Code to terminal - 036 031
- Function - Releases keyboard from locked condition. The keyboard is locked while the terminal is responding to host supplied instructions, but unlocks itself after completion of the instruction. During buffered operation, the keyboard is locked whenever a delimiter key is pressed and must be unlocked by an instruction from the host. When the keyboard is locked, data entry from the keyboard is prohibited and the bell sounds with each keystroke.
- Terminal Response - 036 176 057

SET INTERACTIVE MODE
- Code to terminal - 036 000
- Function - Removes terminal from buffered mode and places it in interactive mode. Upon power-up or master reset, terminal automatically goes into interactive mode, roll enabled. If page mode, non-scrolling function, is desired it must be set by the host.
- Terminal Response - 036 176 057

SET BUFFERED MODE

Code to terminal - 036 001

Function - Puts terminal into buffered page mode. While in buffered mode, keyboard entry is blocked from the host except for the programmed delimiter codes which flag the host to service the terminal. The Roll Mode (screen scrolling) cannot be used in the buffered mode.

READ STATUS

Code to terminal - 036 024

Function - Reads contents of the six status registers in the terminal back to the host in the following order:

- terminal mode
- keyboard lock
- tab mode
- replace tab mode
- tab interval
- fill character

Terminal Response - 036 176 045 <byte 1>....<byte 6> 036 176 057

BELL

Code to terminal - 007 (6053 compatible)

Function - A short audible tone is produced on the speaker located in the terminal.

Screen Control

ROLL ENABLE

Code to terminal - 022 (6053 compatible)

Function - The display is roll enabled; that is, the screen rolls up one line each time either the cursor overflows the bottom line or a New Line character is decoded when the cursor is on the bottom line. When this occurs, the cursor moves to the first character position on the bottom line, the bottom line becomes blank, and the information previously displayed on the top line is lost.

ROLL DISABLE

Code to terminal - 023 (6053 compatible)

Function - The roll enable mode is terminated. When the display is not roll enabled, the cursor moves to the home position each time either the cursor overflows the bottom line or a New Line code is decoded when the cursor is on the bottom line. While the cursor position changes, the information displayed on the screen remains unchanged.

READ SCREEN

Code to terminal - 036 014 <mask> <word> <XE> <YE>

Function - Causes the selected screen data to be read back to the host. The reading starts at the current cursor position and ends at the defined XE (column) and YE (row) screen coordinates.

For a character to be read, the attribute byte when ANDed with the "mask" byte must match the "word" byte. To read all data irrespective of attributes, set both the mask byte and word byte to zero. Then all the data on the screen from cursor location to ending coordinates will be sent.

If the ending coordinates are less than the cursor location, the terminal response will define the current cursor location but will not contain any data. If the the ending coordinates exceed the screen ending, then the terminal response will show data only up to the screen ending.

When the read screen operation skips locations due to improper attributes, the starting address of the new location will be read before the subsequent data string.

Terminal response - 037 176 040 <037 X Y><data string><037 X Y> <data string>..... 036 176 057

SET DELIMITER TABLE

Code to terminal: 036 020 <byte 1><byte 2>.....etc. 036 176 057

Function - Clears the Delimiter Table in the terminal and writes in the new delimiters. These remain in effect until changed by a new command from the host, or a master reset is received, or a new power-up cycle occurs. In the case of the last two, the default delimiters are NEW LINE and CR. Up to 256 delimiter codes can be defined.

Terminal response - 036 176 057

READ DELIMITER TABLE

Code to terminal - 036 023

Function - Causes the delimiter codes as defined by the Delimiter Table to be read back to the host.

Terminal response - 036 176 044 <byte 1><byte 2>.....036 176 057

SET FILL CHARACTER

Code to terminal - 036 022 <byte>

Function - Loads the character code into the Fill Character Register in the terminal. This code is retained until changed by a new command from the host, a master reset, or a new power-up cycle. In the case of the last two, the register is set to the Sysgen defined character.

Terminal response - 036 176 057

FILL UNPROTECTED FIELDS

Code to terminal - 036 003

Function - Causes all unprotected fields to be filled with the fill character. If there are no protected fields, then the whole screen is filled. The cursor is positioned at the first unprotected location on the screen. Attributes of the fill characters are set according to contents of the current attribute register in the terminal. CAUTION; use this function only in the page mode as continuous scrolling will occur if used in the scrolling mode.

Terminal response - 036 176 057

FILL CHARACTER STRING

Code to terminal - 036 013 <no. of locations> <insert character>

Function - Causes the specified number of locations starting at cursor to be replaced with the insert character. If the specified number of locations exceed the line, then the fill operation terminates at line end. Cursor position remains unchanged. Attributes of the inserted characters are set to the current attributes.

INSERT CHARACTER

Code to terminal - 036 010 <no. of characters> <insert Character>

Function - Causes the specified number of characters to shift right one location and the insert character be inserted at cursor location. The last character of the string is deleted. If the number of characters specified exceeds the line, the operation terminates at line end. Attributes of the inserted character are set to the current attributes. Cursor remains at starting position.

BLOCK MOVE/FILL

Code to terminal - 036 006 <XD><YD><XE><YE>

where: <XD><YD> = <COL><ROW> for destination of move
<XE><YE> = <COL><ROW> for end location minus one of string to be moved.

Function - Causes block of data comprising the characters from cursor location to defined end location minus one to be moved to defined destination. If the destination is after the cursor position, the block moves to the right and downward on the screen. Conversely, if the destination is before the cursor location, the block moves left and upward on the screen. In either case, the vacated spaces are loaded with the fill character and the cursor is positioned at the first fill character location.

Terminal response - 036 176 057

DELETE CHARACTER

Code to terminal - 036 011 <no. of characters>

Function - Causes the specified number of characters starting at cursor location to shift left one position. In this operaton the character at cursor location is deleted and a fill character is inserted in the vacated location at end of string. If number of specified characters exceed the line end, the operation terminates at line end.

ERASE TO END OF LINE

Code to terminal - 013

Function - Clears the screen from cursor location to end of line.

ERASE PAGE

Code to terminal - 014

Function - Clears the screen and moves the cursor to the home position. Additionally, this command sets the Blink Enable flag, and clears the Blink, Dim, and Underscore attributes in the Current Attribute Register.

Attribute Control

SET CURRENT ATTRIBUTE REGISTER

Code to terminal - 036 017 <mask byte> <XOR byte>

Function - Sets the Current Attribute Register in the terminal to desired attribute configuration. See the masking/XOR description above. Note that the blink attribute is not functional unless the Blink Enable flag is set in the terminal. Also, an Erase Page command from the host will clear the register, therefore reconfiguration will be required. This instruction can be used instead of the individual set and clear instructions (016, 017, 024, 025, 034, 035).

Terminal response - 036 176 057

READ CURRENT ATTRIBUTE REGISTER

Code to terminal - 036 016

Function - Reads contents of the Current Attribute Register in the terminal and transfers the information back to the host.

Terminal response - 036 176 042 <register byte> 036 176 057

READ ATTRIBUTE STRING

Code to terminal - 036 015 <XE> <YE>

Function - Reads the attributes of a string of characters starting at cursor location and ending at the specified XE (column) and YE (row) coordinates of the screen. The cursor stop position is at the XE and YE ending coordinates.

Terminal response - 036 176 041 <byte><byte>.....036 176 057

SET ATTRIBUTE STRING

Code to terminal - 036 002 <# chars.> <mask byte> <XOR byte> 036 176 057

Function - Redefines the attributes of a specified number of display characters starting at cursor location. If the number of characters specified is 0, then the whole screen will be affected. See description of masking/XOR functions under heading: Character Attributes.

Terminal response - 036 176 057

ENABLE BLINK

Code to terminal - 003 (6053 compatible)

Function - Each character whose blink attribute bit is set is blinked on the screen.

DISABLE BLINK

Code to terminal - 004 (6053 compatible)

Function - None of the characters displayed on the screen are blinked.

START BLINK

Code to terminal - 016 (6053 compatible)
Function - Sets the Blink Attribute bit in the Current Attribute Register. This sets the blink attribute bit of each succeeding character as it is received. These characters blink if the screen is blink enabled (Blink Enable flag set).

END BLINK

Code to terminal - 017 (6053 compatible)
Function - Clears the Blink Attribute bit in the Current Attribute Register. This clears the blink attribute bit on each succeeding character as it is received.

START DIM

Code to terminal - 034 (6053 compatible)
Function - Sets the Dim attribute bit in the Current Attribute Register. This sets the Dim attribute bit of each succeeding character as it is received. These characters are displayed at reduced intensity (dimmed).

END DIM

Code to terminal - 035 (6053 compatible)
Function - Clears the Dim attribute bit in the Current Attribute Register. This clears the Dim attribute bit of each succeeding character as it is received.

START UNDERSCORE

Code to terminal - 024 (6053 compatible)
Function - Sets the Underscore attribute bit in the Current Attribute Register. This sets the Underscore attribute bit on each succeeding character as it is received. These characters are displayed with an underscore.

END UNDERSCORE

Code to terminal - 025 (6053 compatible)
Function - Clears the Underscore attribute bit in the Current Attribute Register. This clears the Underscore attribute bit on each succeeding character as it is received.

Cursor Positioning

HOME

Code to terminal - 010 (6053 compatible)
Function - The cursor moves to the first (leftmost) character position on the top line of the screen, which is the cursor home position.

NEW LINE

Code to terminal - 012 (6053 compatible)
Function - The cursor moves to the first character position on the next line of the screen. If the cursor is on the bottom line, it moves to the home position, unless the terminal is roll enabled (refer to the ROLL ENABLE command described below).

CARRIAGE RETURN
Code to terminal - 015 (6053 compatible)
Function - The cursor moves to the first character position on the line on which the cursor resides.

WRITE CURSOR ADDRESS
Code to terminal - 020 <X> <Y> (6053 compatible)
Function - The display is forced to use the next two codes received as the cursor's new column and row (line) addresses. Both the columns and rowa are numbered beginning with column 0, line 0. The columns are numbered from left to right across the screen; the rows are numbered from top to bottom. After the second character is received, the cursor moves to the location on the screen specified by the new coordinates.

READ CURSOR ADDRESS
Code to terminal - 005 (6053 compatible)
Function - A sequence of three codes is sent from the display to the host computer. The first code is the ASCII control character, Unit Separator (037); the second is the cursor's current 7-bit column address; and the third is the cursor's current 5-bit line address.
Terminal Response - 037 <col.> <row>

CURSOR UP
Code to terminal - 027 (6053 compatible)
Function - The cursor moves up one line while remaining in the same column position. If the cursor is on the top line, it moves to the bottom line of the screen.

CURSOR RIGHT
Code to terminal - 030 (6053 compatible)
Function - The cursor moves one character (column) position to the right. If the cursor is at the end of the line, a New Line operation is performed.

CURSOR LEFT
Code to terminal - 031 (6053 compatible)
Function - The cursor moves one character position to the left. It the cursor is in the leftmost position on the line, it moves to the rightmost position and then up one line.

CURSOR DOWN
Code to terminal - 032 (6053 compatible)
Function - The cursor moves down one line while remaining in the same column position. If the cursor is on the bottom line, it moves to the top line.

Tab Control

SET TAB INTERVAL
Coding to terminal - 036 021 <byte>
Function - Set tab interval in terminal to value of <byte>.
Terminal response - 036 176 057

TAB TO NEXT TAB STOP
Code to terminal - 011
Function - Moves cursor to next tab stop. To actually store a tab code, use the Insert Character instruction to insert the 011 code.
Terminal response - 036 176 057

TAB TO NEXT UNPROTECTED FIELD
Code to terminal - 036 004
Function - Moves cursor to first location in next unprotected field. If used in last field of screen in roll mode, will result in continuous scrolling. If in page mode, the same starting location will result in a cursor move to the first unprotected field at top of screen.
Terminal response - 036 176 057

TAB TO PREVIOUS UNPROTECTED FIELD
Code to terminal - 036 005
Function - Moves cursor to first character location of previous unprotected field.
Terminal response - 036 176 057

SEND TAB CHARACTERS
Code to terminal - 036 030
Function - Used when tabbed data is to be read back to the host; causes space characters to be flagged (bit 0 of character byte set) and a tab character substituted in their place.
Terminal response - 036 176 057

REPLACE TAB CHARACTERS
Code to terminal - 036 027
Function - The opposite effect of the SEND TAB CHARACTERS instruction. Causes the individual space codes to be sent back to the host instead of the substitute tab character.
Terminal response - 036 176 057

Printer Control

PRINT
Code to terminal - 021 (6053 compatible)
Keying - PRINT key (generates 036 021 code to host)
Function - Data displayed on the screen, beginning with the leftmost character on the cursor line and ending with the rightmost character on the bottom line, is printed. During the print operation, the keyboard is disabled.

PRINT FORM

Code to terminal - 001 (6053 compatible)
Keying - SHIFT PRINT (generates 036 001 to host)
Function - If terminal Sysgen is configured for 6053 compatibility, then this command will execute a print-out of screen data as described for the PRINT command except only the full intensity characters will be printed. All dim characters will be treated as spaces. If Sysgen is not configured for 6053 compatibility, the unprotected characters are sent to the printer.

PRINT LINE

Coding to terminal - 036 007 <data string><terminator>
Function - The data string is sent to the terminal for transfer to the printer. Each data string can contain up to 132 characters including the terminator. Character strings in excess of 132 will be truncated. The terminator can be any one of the following:

terminator	codes to printer	
015	015	(carr. ret.)
012	015 012	(carr. ret. + new line)
014	015 014	(carr. ret. + form feed - selects new page)
000	000	(null - stops printing at current location)

Note that this instruction does not affect data on the screen, but does lock the keyboard while the data is being transferred.
Terminal response - 036 006

Extended Graphic Character Set

Individual characters of the 32-character Extended Graphic Character Set can be accessed by host instructions with the following two byte format:

033 <character code>

Refer to Appendix D for character codes and associated display symbols.

Since the code for these characters include octal numbers 0-37, some care is required in using the Read Screen instruction when they are in use. To avoid confusion with the cursor position, use zero for both the XOR byte and word byte of the Read Screen instruction. Then the only cursor position that will be received is the first one of the terminal response. The host program can be structured to treat subsequent 037 codes as data.

BUFFERED MODE PROGRAMMING

Basically, programming for buffered mode operation has two concerns:

1. Preparation of the terminal for buffered operation
2. Host response to delimiter codes from the terminal.

In the first of these, preparing the terminal for buffered operation, the following operations are essential:

*Write any required screen format
*Define protected and unprotected fields
*Condition other character attributes as required
*Define delimiters
*Define Fill character, if required
*Define Tab interval, if required
*Position cursor at starting location
*Set buffered mode

The host response to delimiter codes has the following concerns:

*Set interactive mode (to enable communication to terminal
*Read screen data
*Erase screen, either completely or by unprotected field
*Write new screen format, if required
*Unlock keyboard
*Set buffered mode

Note that the response to different delimiter codes can be programmed for functions other than a Read Screen operation.

The following programming examples show how the terminal can be prepared for two types of buffered mode operations. In the first example (see Figure 2 and Table 1), a sample format is written to the screen with protected and unprotected fields. Protected form areas are programmed for dimmed display thereby giving contrast to unprotected fields and the data entered by the operator. The areas where the operator is expected to enter data is also identified by underscoring. The fill character is defined as a space code.

The second programming example (see Figure 3 and Table 2) shows how a tabular application could be implemented. In this, a simple heading format is written to the screen and is protected. The tab interval is defined to give four identical columns.

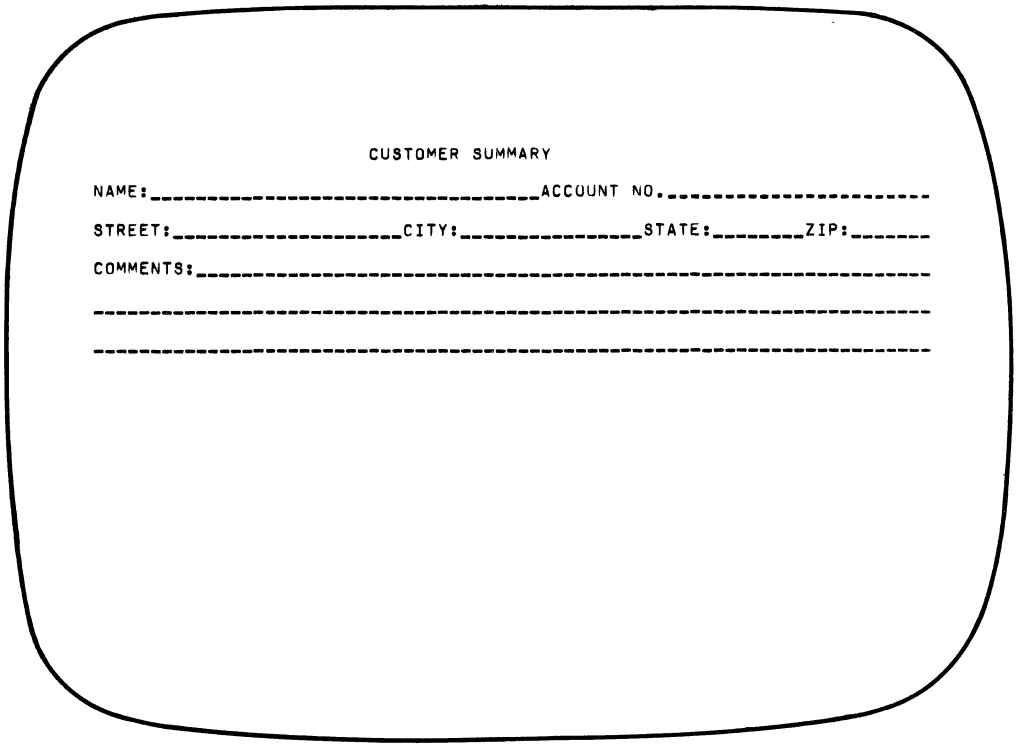


Figure 2 Sample Screen For Protected/Unprotected Field Format

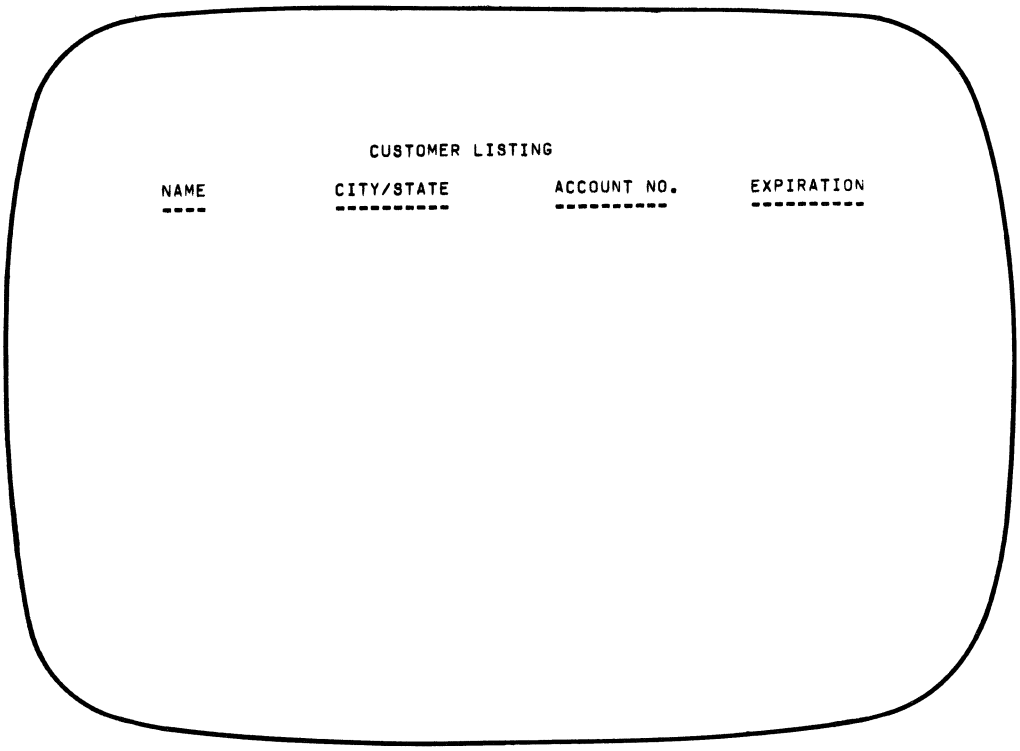


Figure 3 Sample Screen For Tabular Field Format

Table 1 Sample Program For Protected/Unprotected Field Format

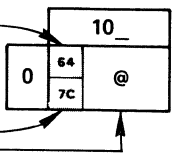
CODING (octal)	FUNCTION
036 025	Reset terminal
023	Disable Roll
020 030 004	Move cursor
036 017 000 040	Set Protect attribute
103 125 123 124 117 115 105 122 040	write: CUSTOMER SUMMARY
123 125 115 115 110 122 131	
020 000 006	Move cursor
036 017 040 010	Set Dim; maintain Protect
116 110 115 105 072	Write: NAME:
020 047 006	Move cursor
110 103 103 117 125 116 124 040 116	Write: ACCOUNT NO.:
117 056 072	
020 000 010	Move cursor
123 124 122 105 105 124 072	Write: STREET:
020 034 010	Move cursor
103 111 124 131 072	write: CITY:
020 060 010	Move cursor
123 124 110 124 105 072	write: STATE:
020 075 010	Move cursor
132 111 120 072	write: ZIP:
020 000 012	Move cursor
103 117 115 115 105 116 124 123 072	write: COMMENTS:
020 000 017	Move cursor
036 002 000 000 040	Set Protect for rest of screen
036 002 177 000 040	Set Protect for blank areas
036 002 177 000 040	Set Protect for blank areas
036 002 131 000 040	Set Protect for blank areas
020 050 004	Move cursor
036 002 170 000 040	Set Protect for blank areas
020 000 007	Move cursor
036 002 120 000 040	Set Protect for blank areas
020 000 011	Mov cursor
020 002 120 000 040	Set Protect for blank areas
020 000 013	Move cursor
036 002 120 000 040	Set Protect for blank areas
020 000 015	Move cursor
036 002 120 000 040	Set Protect for blank areas
036 017 000 020	Set Underscore; reset Protect
036 022 040	Set Fill Character = space
036 003	Fill unprotected fields
020 005 006	Move cursor
036 020 036 022 036 176 057	Set delimiter = SEND key
036 031	Unlock keyboard
036 001	Set Buffered Mode

Table 2 Sample Program For Tabular Field Format

CODING (octal)	FUNCTION
036 025	Reset terminal
020 032 004	Move cursor
036 017 000 040	Set Protect attribute
103 125 123 124 117 115 105 122 040	Write: CUSTOMER LISTING
114 111 123 124 111 116 107	
020 010 006	Move cursor
036 017 000 060	Set Protect/Underscore attributes
116 110 115 105	Write: NAME
020 027 006	Move cursor
103 111 124 131 057 123 124 110 124	Write: CITY/STATE
105	
020 052 006	Move cursor
110 103 103 117 125 116 124 040 116	Write: ACCOUNT NO.
117 056	
020 073 006	Move cursor
105 130 120 111 122 110 124 111 117	Write: EXPIRATION
116	
020 000 010	Move cursor
036 017 000 000	Clear attributes
036 021 024	Set Tab Interval = 20 (decimal)
036 020 036 022 036 176 057	Set delimiter = SEND key
036 022 040	Set Fill Character = space
036 031	Unlock keyboard
036 001	Set Buffered Mode

APPENDIX A

ASCII CHARACTER CODES AND TERMINAL CONTROL FUNCTIONS



↥ *means* **CONTROL**

OCTAL	00_		CONTROL FUNCTION	01_		CONTROL FUNCTION	02_		CONTROL FUNCTION	03_		CONTROL FUNCTION	04_		05_	
0	0 00	NUL	NULL	8 16	BS *(BACK-SPACE)*	HOME	16 10	DLE ↥P	WRITE CURSOR HEADER	24 18	CAN ↥X	CURSOR RIGHT	32 40	SPACE	40 4D	(
1	1 01	SOH ↥A	PRINT FORM	9 05	HT *(TAB)*	HORIZONTAL TAB	17 11	DC1 ↥Q	PRINT PAGE	25 19	EM ↥Y	CURSOR LEFT	33 5A	!	41 5D	)
2	2 02	STX ↥B	START OF TEXT	10 15	NL *(NEW LINE)*	NEW LINE	18 12	DC2 ↥R	ROLL ON	26 3F	SUB ↥Z	CURSOR DOWN	34 7F	,, *(QUOTE)*	42 5C	*
3	3 03	ETX ↥C	ENABLE BLINK	11 0B	VT *(VERT. TAB)*	ERASE LINE	19 13	DC3 ↥S	ROLL OFF	27 27	ESC *(ESCAPE)*	ESCAPE	35 7B	#	43 4E	+
4	4 37	EOT ↥D	INHIBIT BLINK	12 06	FF *(FORM FEED)*	ERASE PAGE	20 3C	DC4 ↥T	UNDERSCORE ON	28 1C	FS ↥\	DIM ON	36 5B	$	44 6B	, *(COMMA)*
5	5 2D	ENQ ↥E	CURSOR ADDRESS READ	13 0D	RT *(RETURN)*	RETURN	21 3D	NAK ↥U	UNDERSCORE OFF	29 1D	CS ↥]	DIM OFF	37 6C	%	45 60	–
6	6 2E	ACK ↥F	PRINT DONE	14 0E	SO ↥N	BLINK ON	22 32	SYN ↥V	SYNCHRONOUS IDLE	30 1E	RS ↥↑	FUNCTION KEY HEADER	38 50	&	46 4B	. *(PERIOD)*
7	7 2F	BEL ↥G	BELL	15 0F	SI ↥O	BLINK OFF	23 26	ETB ↥W	CURSOR UP	31 1F	US ↥←	READ CURSOR HEADER	39 7D	, *(APOS)*	47 61	/

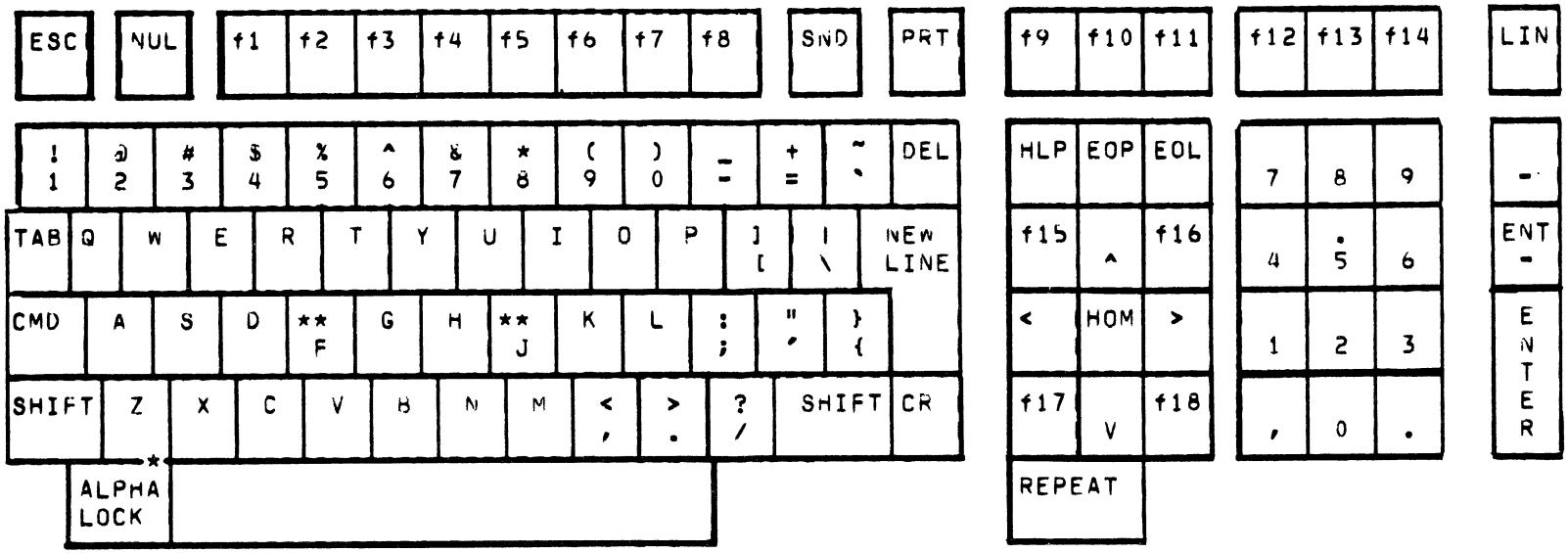
OCTAL	06_		07_		10_		11_		12_		13_		14_		15_		16_		17_	
0	48 F0	Ø	56 F8	8	64 7C	@	72 C8	H	80 D7	P	88 E7	X	96 79	` *(GRAVE)*	104 88	h	112 97	p	120 A7	x
1	49 F1	1	57 F9	9	65 C1	A	73 C9	I	81 D8	Q	89 E8	Y	97 81	a	105 89	i	113 98	q	121 A8	y
2	50 F2	2	58 7A	:	66 C2	B	74 D1	J	82 D9	R	90 E9	Z	98 82	b	106 91	j	114 99	r	122 A9	z
3	51 F3	3	59 5E	;	67 C3	C	75 D2	K	83 E2	S	91 8D	[	99 83	c	107 92	k	115 A2	s	123 C0	{
4	52 F4	4	60 4C	<	68 C4	D	76 D3	L	84 E3	T	92 E0	\	100 84	d	108 93	l	116 A3	t	124 4F	|
5	53 F5	5	61 7E	=	69 65	E	77 D4	M	85 E4	U	93 9D	]	101 85	e	109 94	m	117 A4	u	125 D0	}
6	54 F6	6	62 6E	>	70 C6	F	78 D5	N	86 E5	V	94 5F	↑ or ^	102 86	f	110 95	n	118 A5	v	126 A1	~ *(TILDE)*
7	55 F7	7	63 6F	?	71 C7	G	79 D6	O	87 E6	W	95 6D	← or_	103 87	g	111 96	o	119 A6	w̄	127 07	DEL *(RUBOUT)*

CHARACTER CODE IN OCTAL AT TOP AND LEFT OF CHARTS.

DG-03384

APPENDIX B

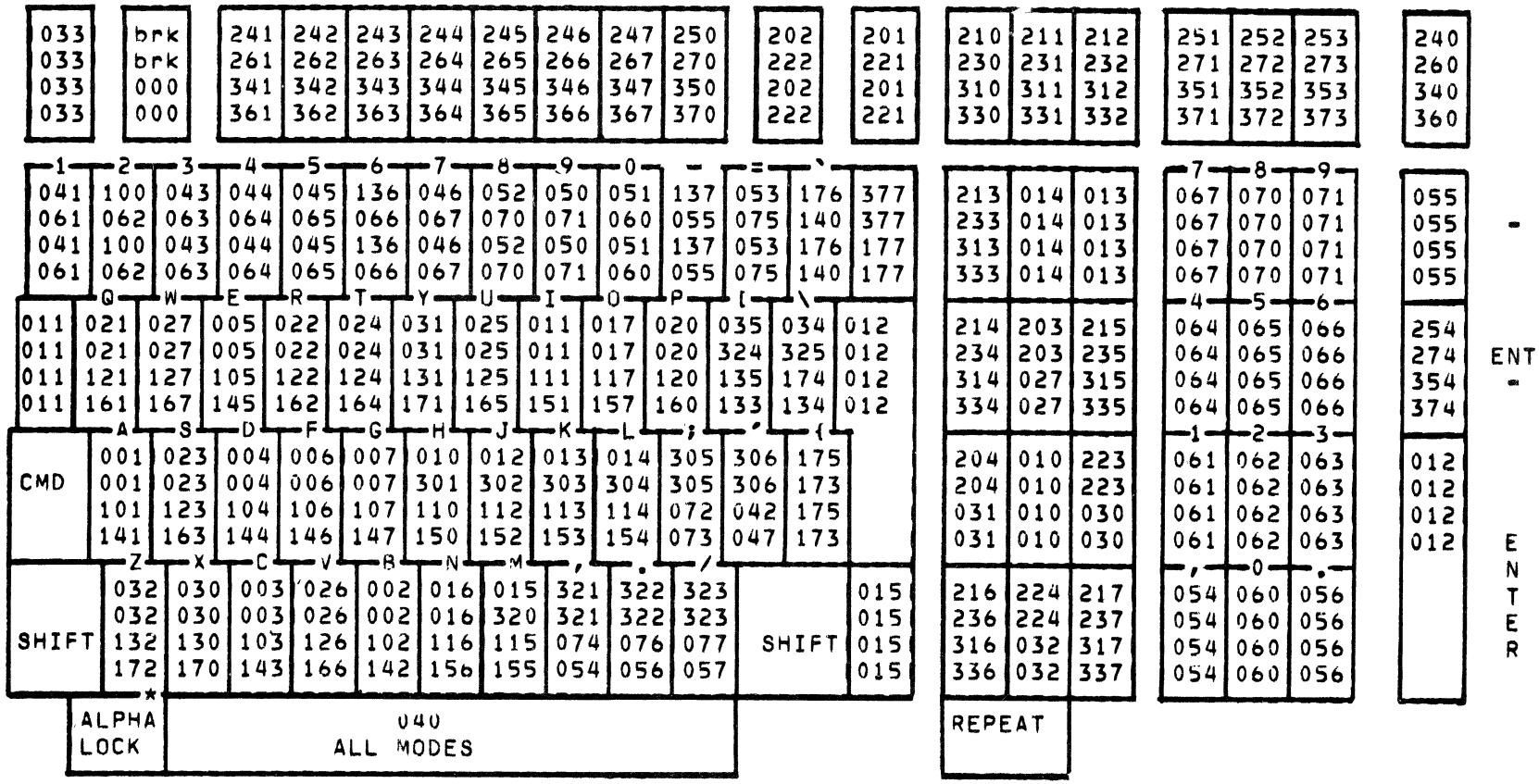
KEYBOARD LAYOUT



* LIGHTED KEY

APPENDIX C

KEYBOARD CODE ASSIGNMENTS



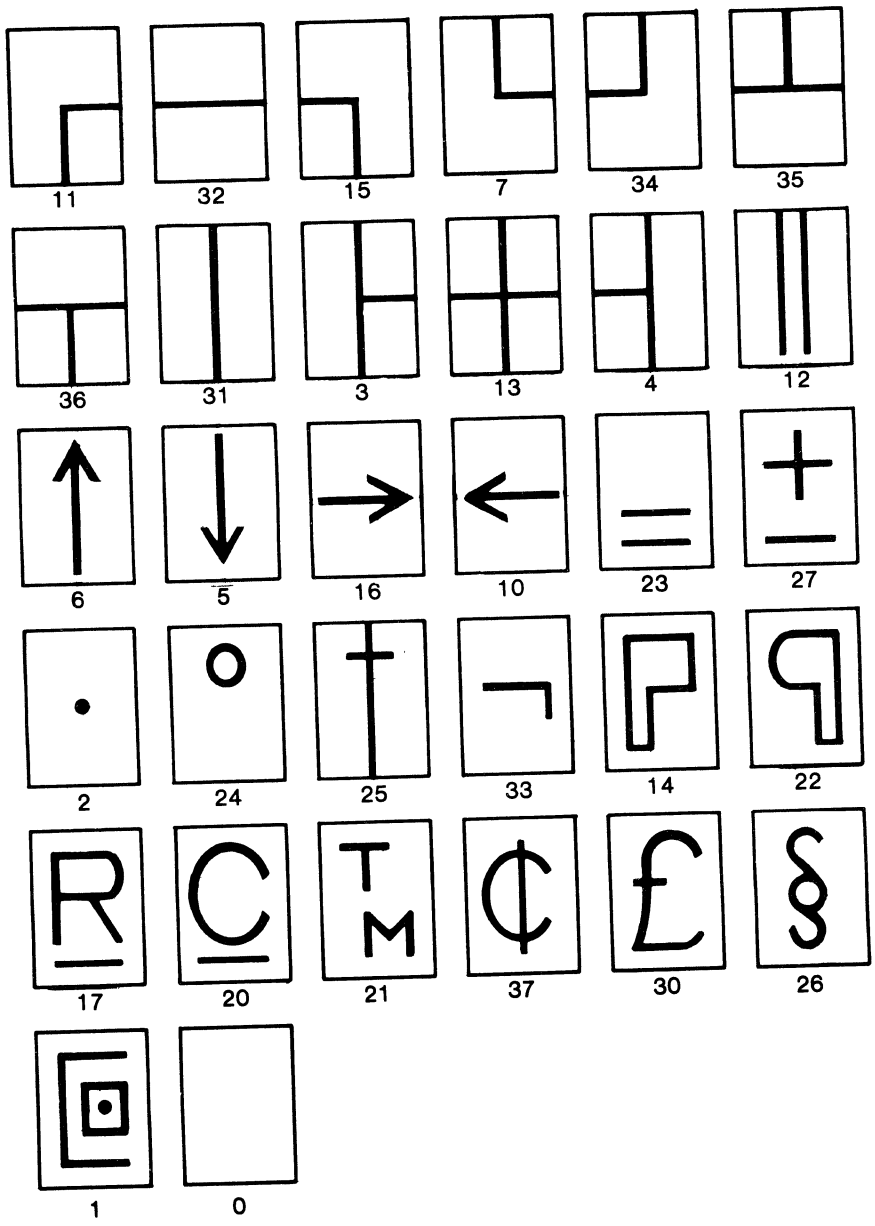
CODE LEGEND

...LED

Note - The maximum 3-digit octal code value that can transfer between host and terminal is 177. Values larger than this, go in a two code format: 036 XXX. The 036 carries a value of 200 octal which added with value of the second code completes the code value indicated on the code chart.

APPENDIX D

EXTENDED GRAPHIC SYMBOLS AND CODES



Note - When coding for the extended graphic symbols, the indicated codes must be preceded by an 033 code.

APPENDIX E

SYSGEN PARAMETERS

INTRODUCTION

Basic initialization parameters which configure the terminal to specific operational requirements of the system are stored in the Sysgen memory area of the terminal. In general, it is expected that these parameters would be defined at the initial setup of the system and would only be changed if reconfiguration of system operation is desired. Conceivably, the responsibility for programming the Sysgen memory would be restricted to system supervisory personnel.

TO CHANGE PARAMETERS

*Select local mode (press CMD and LINE keys simultaneously)

*Press CMD SHIFT LINE keys simultaneously

This action causes the Sysgen parameters to be displayed on the screen as shown in Figure 4. Note that the current parameter values are displayed in the CURRENT column and the cursor is initially in the top field of the NEW column. A new value for a parameter may be inserted at the current cursor position. Only the symbols listed are accepted. A NEW LINE keystroke must terminate each entry. If the entry is not acceptable, the cusor advances to the next field. If NEW LINE is pressed without any entry being made, the current parameter value is retained.

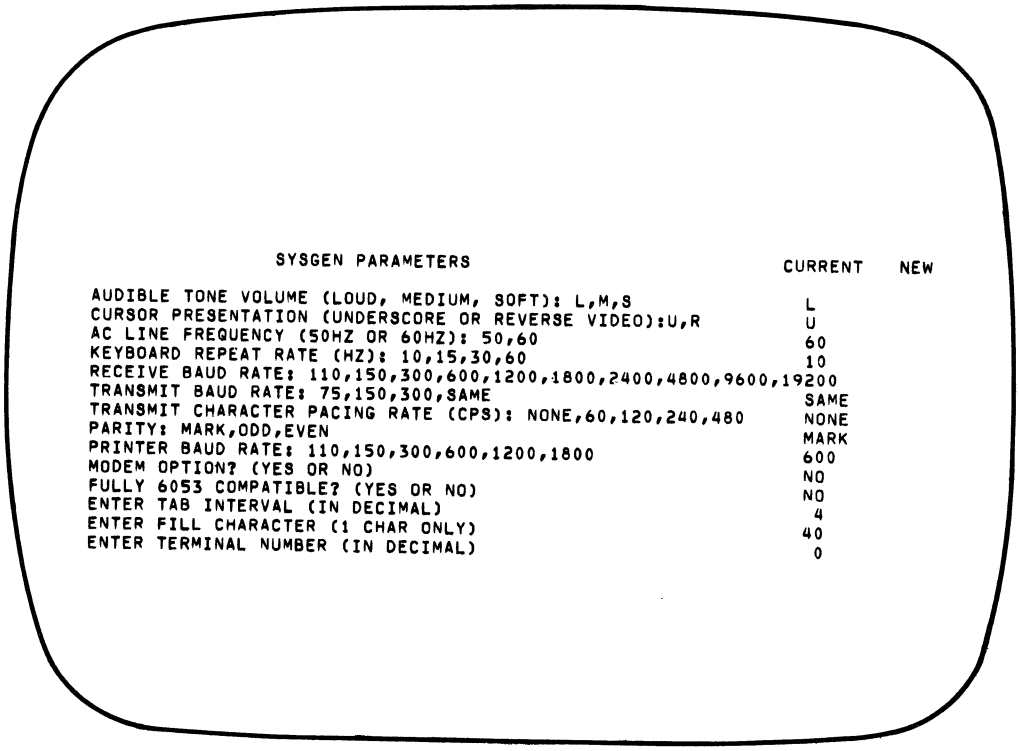


Figure 4 Sysgen Screen Presentation

If the NEW LINE key is pressed while the cursor is in the last field or any time the ESC key is pressed, the screen is erased and the Sysgen memory is closed against new entry. The terminal will now re-initialize itself with the new parameters and will stay in the local mode (press LINE key to get back on-line to host).

In the event a STARTUP ERROR message is displayed on initialization, the number following the error message indicates to service personnel the type of malfunction. The terminal can usually be operated, even if an error is indicated; however, such operation requires that any modification to the default parameters be re-entered each time power is turned on.

ERRATA TO DOCUMENT 055-203-00

DOCUMENT TITLE: DASHER D3 Display Terminal Programmer's Reference Manual (Preliminary)

CHANGES:

Page 4, after paragraph 1, insert the following:

CAUTION

When operating in the bufffered mode, take care not to press the NULL key as this will lock up the terminal. To recover from such a situation, turn power off and then on again.

Page 13, replace the "Protected Data" description with the following:

Protected Data - The Protected Data attribute is controlled by bit 2 of the character's attribute byte.

In buffered mode, the operator is not allowed to modify any character which has its Protected Data attribute bit set. This is accomplisned primarily by not allowing the cursor to be positioned in any protected area of the screen.

In interactive mode, the cursor can be positioned any place regardless of the protected status of a character. This allows the screen to be formatted in preparation for entering buffered mode.

The correct procedure for modifing a protected character is to reset the Protected Data attribute bit to 0, modify the chatacter and then set the Protected Data attribute bit to 1, if desired.

Any software manipulating screen data in interactive mode should also keep track of the cursor position relative to protected data, as output could be lost in these areas. Also, some character sequences could modify unwanted areas of the screen.

Page 15, after paragraph 2, insert the following:

Program Timing

Some complex commands require that the terminal's internal processor execute extended routines (e.g. the ERASE PAGE command). At high baud rates, the execution time of these commands will exceed the minimum transmission periods from the host. All characters received from the host, both command and data, are internally buffered in the terminal.

This buffering is sufficient to prevent loss of data or commands under any rational operating condition.

It is possible, however, to deliberately cause the loss of data by repeating complex commands at high baud rates. Two examples are:

> At 19.2 Kbaud, if more than six ERASE PAGE commands were transmitted successively followed by a long data stream, some of the data could be lost.
>
> At 19.2 Kbaud, in roll mode, if a sequence of NEW LINE followed by one or two data characters were repeated several hundred times, data could eventually be lost (in this example, the screen would be scrolling at about 500 lines per second).

If for some unusual reason, such a sequence of complex commands is required, the programmer must insert appropriate delays.