

**SPERRY UNIVAC**  
**UNISCOPE**  
**Display Terminal**

**Programmer Reference**

This document contains the latest information available at the time of publication. However, Sperry Univac reserves the right to modify or revise its contents. To ensure that you have the most recent information, contact your local Sperry Univac representative.

Sperry Univac is a division of Sperry Rand Corporation.

SPERRY UNIVAC and UNISCOPE are trademarks of the Sperry Rand Corporation.

PAGE STATUS SUMMARY

ISSUE: UP-7807 Rev. 2

Part/Section	Page Number	Update Level
Cover/ Disclaimer		
PSS	1	
Contents	1 thru 8	
1	1 thru 3	
2	1 thru 40	
3	1 thru 35	
4	1 thru 23	
Appendix A	1 thru 3	
Appendix B	1 thru 3	
Appendix C	1 thru 9	
Appendix D	1 thru 25	
Index	1 thru 20	
User Comment Sheet		
Total 173 pages including covers		

Part/Section	Page Number	Update Level

Part/Section	Page Number	Update Level



# Contents

## PAGE STATUS SUMMARY

## CONTENTS

### 1. INTRODUCTION

1.1.	GENERAL . . . . .	1-1
1.2.	USER DOCUMENTS . . . . .	1-2
1.3.	TERMINAL CONFIGURATIONS . . . . .	1-2

### 2. FUNCTIONAL DESCRIPTION

2.1.	GENERAL CHARACTERISTICS OF UNISCOPE DISPLAY TERMINAL . . . . .	2-1
2.2.	OPERATION OF UNISCOPE DISPLAY TERMINAL . . . . .	2-3
2.2.1.	Cathode-Ray Tube Display Screen . . . . .	2-3
2.2.2.	Cursor . . . . .	2-4
2.2.3.	Keyboard Operation . . . . .	2-4
2.2.3.1.	Alphanumeric/Symbolic Section . . . . .	2-5
2.2.3.2.	Numeric Section . . . . .	2-5
2.2.3.3.	Message Control Section . . . . .	2-7
2.2.3.4.	Cursor Control Section . . . . .	2-7
2.2.3.5.	Editing Control Section . . . . .	2-8
2.2.3.6.	Special Function Section . . . . .	2-8
2.2.3.7.	Protected Format Section . . . . .	2-8
2.2.4.	Control and Indicator Panel . . . . .	2-9
2.2.5.	Message Buffering . . . . .	2-10
2.2.6.	Keyboard Lock Function . . . . .	2-10
2.2.7.	Processor-Controlled Operation . . . . .	2-10
2.2.7.1.	Request Retransmission . . . . .	2-11
2.2.7.2.	Field Blinking . . . . .	2-11
2.2.7.3.	Line Insert . . . . .	2-11
2.2.7.4.	Line Delete . . . . .	2-11
2.2.7.5.	Display Rolling . . . . .	2-12
2.2.7.6.	Horizontal Tab . . . . .	2-12
2.2.7.7.	Cursor Return/Line Feed . . . . .	2-12
2.2.7.8.	MESSAGE WAIT Indicator . . . . .	2-12

2.3.	TERMINAL MULTIPLEXER CONNECTIONS . . . . .	2-12
2.4.	COMMUNICATIONS INTERFACE BETWEEN UNISCOPE TERMINAL AND PROCESSOR . . . . .	2-13
2.4.1.	Direct Connection . . . . .	2-13
2.4.2.	Remote Synchronous Operation (SPERRY UNIVAC Processors) . . . . .	2-14
2.4.3.	Remote Asynchronous Operation (SPERRY UNIVAC Processors). . . . .	2-16
2.4.4.	Remote Synchronous Operation (IBM System/360 or System/370) . . . . .	2-17
2.4.5.	Remote Asynchronous Operation (IBM System/360 or System/370) . . . . .	2-18
2.5.	COMMUNICATIONS SEQUENCE . . . . .	2-19
2.5.1.	Data Flow for Messages to Terminal . . . . .	2-19
2.5.2.	Data Flow for Messages from Terminal . . . . .	2-19
2.5.3.	Message Format . . . . .	2-19
2.5.4.	Message Acknowledgment . . . . .	2-20
2.5.5.	Keyboard Functions and Octal Codes . . . . .	2-21
2.6.	UNISCOPE TERMINAL AUXILIARY INTERFACE . . . . .	2-21
2.6.1.	Functional Operation . . . . .	2-22
2.6.1.1.	Selection . . . . .	2-22
2.6.1.2.	Device Identifier (DID) . . . . .	2-22
2.6.1.3.	DID Register . . . . .	2-22
2.6.1.4.	Auxiliary Interface Input Enable . . . . .	2-22
2.6.1.5.	Auxiliary Interface Output Enable . . . . .	2-22
2.6.1.6.	PRINT Key. . . . .	2-24
2.6.1.7.	Print Code . . . . .	2-24
2.6.1.8.	End-of-Memory Output. . . . .	2-24
2.6.1.9.	End-of-Memory Input . . . . .	2-24
2.6.1.10.	Print-Transparent Code . . . . .	2-24
2.6.2.	Operation . . . . .	2-24
2.6.2.1.	Online Operation . . . . .	2-24
2.6.2.2.	Offline Operation. . . . .	2-27
2.6.2.3.	Combined Operation . . . . .	2-28
2.6.2.4.	DLE 8 . . . . .	2-28
2.6.3.	Device Requirements . . . . .	2-28
2.6.4.	UNISCOPE Terminal Operation Without AI . . . . .	2-29
2.7.	DIFFERENCES IN UNITS . . . . .	2-29
2.7.1.	Differences in UNISCOPE 100 Terminal Series . . . . .	2-29
2.7.2.	Differences in UNISCOPE 200 Terminal Series . . . . .	2-30
2.8.	TAPE CASSETTE SYSTEM CHARACTERISTICS AND OPERATION . . . . .	2-30
2.8.1.	Equipment Description . . . . .	2-31
2.8.1.1.	Standard Cassette System Features . . . . .	2-31
2.8.1.2.	Optional Features . . . . .	2-32
2.8.2.	Cassette System Configuration . . . . .	2-33
2.8.3.	Tape Cassette Interchangeability . . . . .	2-33
2.8.4.	Write/Read Implementation . . . . .	2-34
2.8.4.1.	Tape Format . . . . .	2-34
2.8.4.2.	Selection . . . . .	2-35
2.8.4.3.	Write Operation. . . . .	2-35
2.8.4.4.	Read Operation . . . . .	2-35
2.8.4.5.	Cursor Positioning . . . . .	2-36

2.8.5.	Parity Generation and Checking . . . . .	2-36
2.8.6.	Error Detection, Reporting, and Recovery Operations . . . . .	2-37
2.8.6.1.	Parity Errors . . . . .	2-37
2.8.6.2.	Timing Errors . . . . .	2-37
2.8.6.3.	Improper-Selection Errors . . . . .	2-37
2.8.6.4.	Tape Problems . . . . .	2-38
2.8.6.5.	Read-After-Write Errors . . . . .	2-38
2.8.7.	Search . . . . .	2-38
2.8.7.1.	Mode @ Search . . . . .	2-38
2.8.7.2.	Mode A Search . . . . .	2-39
2.8.7.3.	Mode B Search . . . . .	2-39
2.8.7.4.	Mode C Search . . . . .	2-39
2.8.8.	Backward-One-Block Tape Repositioning . . . . .	2-39
2.8.9.	Address Reporting . . . . .	2-39
2.8.10.	Beginning-of-Tape and End-of-Tape Detection . . . . .	2-40
2.8.11.	Load Point . . . . .	2-40

### 3. CODES

3.1.	GENERAL . . . . .	3-1
3.2.	MESSAGE FORMATS . . . . .	3-1
3.3.	MESSAGE TYPES . . . . .	3-1
3.3.1.	Supervisory Messages . . . . .	3-1
3.3.2.	Messages to Terminal . . . . .	3-2
3.3.2.1.	Polls . . . . .	3-2
3.3.2.2.	Text . . . . .	3-4
3.3.2.3.	Break and Resume (Embedded Messages) - DLE [ and DLE ] . . . . .	3-4
3.3.3.	Messages from Terminal . . . . .	3-4
3.3.3.1.	Text - Terminal to Processor . . . . .	3-4
3.3.3.2.	Supervisory Messages - Terminal to Processor . . . . .	3-6
3.3.4.	Urgent Processor Message . . . . .	3-7
3.3.5.	Interface Timing Considerations . . . . .	3-7
3.3.6.	Control Codes and Sequences . . . . .	3-8
3.4.	ADDRESSING AND ROUTING . . . . .	3-10
3.4.1.	Remote Identifier - RID . . . . .	3-10
3.4.2.	Station Identifier - SID . . . . .	3-10
3.4.3.	Device Identifier - DID . . . . .	3-10
3.4.4.	General Identifier - GID . . . . .	3-10
3.5.	TEXT FORMAT . . . . .	3-14
3.5.1.	Cursor Address . . . . .	3-14
3.5.2.	Text . . . . .	3-16
3.5.2.1.	Data Format, Processor to UNISCOPE Terminal . . . . .	3-16
3.5.2.2.	Data Format, UNISCOPE Terminal to Processor . . . . .	3-17
3.5.3.	Display Rolling (Scroll Effect) . . . . .	3-18
3.5.4.	Tab Stop Setting . . . . .	3-21
3.5.5.	Character Structure and Bit Sequencing . . . . .	3-21
3.5.5.1.	Synchronous Transmission . . . . .	3-22
3.5.5.2.	Asynchronous Transmission . . . . .	3-22

3.5.6.	Parity . . . . .	3-23
3.5.7.	Protected Format . . . . .	3-24
3.6.	<b>CODE DEFINITIONS . . . . .</b>	<b>3-25</b>
3.6.1.	<b>Standard Communications Control Characters . . . . .</b>	<b>3-26</b>
3.6.1.1.	SOH – Start of Heading . . . . .	3-26
3.6.1.2.	STX – Start of Text . . . . .	3-26
3.6.1.3.	ETX – End of Text . . . . .	3-26
3.6.1.4.	EOT – End of Transmission . . . . .	3-26
3.6.1.5.	ENQ – Enquiry . . . . .	3-27
3.6.1.6.	SYN – Synchronous Idle . . . . .	3-27
3.6.1.7.	DLE – Data Link Escape . . . . .	3-28
3.6.2.	<b>Code-Extended Communications Control Characters . . . . .</b>	<b>3-28</b>
3.6.2.1.	WABT – Wait Before Transmit (Busy) . . . . .	3-28
3.6.2.2.	DNAK – Negative Acknowledgment . . . . .	3-28
3.6.2.3.	DBR – Break (Before Embedded Segment) . . . . .	3-28
3.6.2.4.	DRE – Resume (After Embedded Segment) . . . . .	3-29
3.6.2.5.	DENQ – Reply Request . . . . .	3-29
3.6.3.	<b>Miscellaneous Control Characters or Sequences . . . . .</b>	<b>3-29</b>
3.6.3.1.	NUL – Null . . . . .	3-29
3.6.3.2.	SO – Shift Out . . . . .	3-29
3.6.3.3.	SI – Shift In . . . . .	3-29
3.6.3.4.	ESC – Escape . . . . .	3-30
3.6.3.5.	BCC – Block Check Character . . . . .	3-30
3.6.4.	<b>Data Characters . . . . .</b>	<b>3-31</b>
3.6.5.	<b>Editing and Device Control Codes . . . . .</b>	<b>3-31</b>
3.6.5.1.	CR – Cursor Return . . . . .	3-31
3.6.5.2.	ESC a – Erase to End of Display . . . . .	3-31
3.6.5.3.	ESC b – Erase to End of Line . . . . .	3-31
3.6.5.4.	ESC C – Delete in Display . . . . .	3-31
3.6.5.5.	ESC c – Delete in Line . . . . .	3-32
3.6.5.6.	ESC D – Insert in Display . . . . .	3-32
3.6.5.7.	ESC d – Insert in Line . . . . .	3-32
3.6.5.8.	ESC e – Cursor to Home . . . . .	3-32
3.6.5.9.	ESC f – Scan Up . . . . .	3-32
3.6.5.10.	ESC g – Scan Left . . . . .	3-32
3.6.5.11.	ESC h – Scan Right . . . . .	3-32
3.6.5.12.	ESC i – Scan Down . . . . .	3-33
3.6.5.13.	SP – Space . . . . .	3-33
3.6.5.14.	HT – Tab . . . . .	3-33
3.6.5.15.	ESC j – Insert Line . . . . .	3-33
3.6.5.16.	ESC k – Delete Line . . . . .	3-33
3.6.5.17.	BEL (Processor) – Message Waiting . . . . .	3-33
3.6.5.18.	BEL (Terminal) – Request Processor Message . . . . .	3-33
3.6.5.19.	ESC HT – Tab Stop Set . . . . .	3-34
3.6.5.20.	FS – Start-Blink Marker . . . . .	3-34
3.6.5.21.	GS – End-Blink Marker . . . . .	3-34
3.6.5.22.	DC4 or ESC DC4 – Lock Keyboard . . . . .	3-34
3.6.5.23.	DC2 – Print (Initiate Auxiliary Interface Function) . . . . .	3-34
3.6.5.24.	ESC DC2 – Print Transparent (Initiate Auxiliary Interface Function) . . . . .	3-34
3.6.5.25.	DC1 – Transmit, Transmit Unprotected Display . . . . .	3-34

3.6.5.26.	RS — Start of Entry (SOE)	3-35
3.6.5.27.	LF, FF — Line Feed and Form Feed	3-35
3.6.5.28.	ESC DC1 — Transmit Display	3-35
3.6.5.29.	ESC M — Erase Display	3-35
3.6.5.30.	ESC K — Erase Field	3-35
4.	<b>TAPE CASSETTE SYSTEM PROGRAMMER INFORMATION AND CONSIDERATIONS</b>	
4.1.	<b>SELECTION AND STATUS REPORTING</b>	4-1
4.1.1.	Selection Methods	4-1
4.1.2.	Status Reporting	4-2
4.1.3.	Status Code Interpretation	4-2
4.2.	<b>ONLINE WRITE</b>	4-3
4.2.1.	Method	4-3
4.2.2.	Precautions	4-5
4.3.	<b>ONLINE READ</b>	4-5
4.3.1.	Method	4-6
4.3.2.	Precautions	4-10
4.4.	<b>AUTOMATIC TRANSMIT</b>	4-10
4.5.	<b>PROCESSOR CONTROL COMMANDS</b>	4-12
4.5.1.	Backward One Block	4-12
4.5.2.	Report Address	4-13
4.5.3.	Search	4-14
4.6.	<b>USE OF SEARCH COMMANDS</b>	4-16
4.6.1.	Types and Command Formats	4-16
4.6.2.	Special Considerations	4-18
4.7.	<b>END-OF-TAPE PROCEDURES</b>	4-18
4.8.	<b>ERROR CONDITIONS AND RECOVERY PROCEDURES</b>	4-19
4.8.1.	Read Errors	4-19
4.8.2.	Write Errors	4-19
4.8.3.	Read-After-Write Errors	4-20
4.8.4.	Improper-Selection Errors	4-20
4.9.	<b>USE OF OPTIONAL FEATURES</b>	4-21
4.9.1.	Read After Write	4-21
4.9.2.	Protected Format	4-21
4.9.2.1.	Protected Format Translation Characters	4-21
4.9.2.2.	Protected Format Enhancements Provided by Auxiliary Interface 1247-01	4-21
4.9.2.3.	HT (Tab Stop) Control Character Feature	4-22
4.9.3.	Record Separator Writing	4-22
4.9.4.	Identifier Search	4-22

4.9.5.	Optional Features Used Only in Offline Operation . . . . .	4-22
4.9.5.1.	List and Edit . . . . .	4-22
4.9.5.2.	Print Transparent . . . . .	4-23
4.9.5.3.	Copy to Address . . . . .	4-23

## APPENDIXES

### A. UNISCOPE TERMINAL RESPONSES TO POLLS

### B. LINE CONTROL RULES

B.1.	GENERAL . . . . .	B-1
B.2.	HARDWARE RULES . . . . .	B-1
B.3.	SOFTWARE RULES . . . . .	B-2

### C. OPERATION SEQUENCE EXAMPLES

C.1.	GENERAL . . . . .	C-1
C.2.	SINGLE STATION (PERFECT TRANSMISSION) . . . . .	C-1
C.3.	SINGLE STATION WITH AUXILIARY INTERFACE DEVICE (PERFECT TRANSMISSION) . . . . .	C-2
C.4.	MULTISTATION WITH TWO TERMINALS ON A MULTIPLEXER (PERFECT TRANSMISSION) . . . . .	C-4
C.5.	MULTISTATION WITH ACKNOWLEDGMENT PASSED BY MULTIPLEXER (PERFECT TRANSMISSION) . . . . .	C-5
C.6.	MULTISTATION - TWO TERMINALS WITH AUXILIARY INTERFACE DEVICES ON A MULTIPLEXER (PERFECT TRANSMISSION) . . . . .	C-7

### D. ERROR RECOVERY

D.1.	GENERAL . . . . .	D-1
D.1.1.	Alternating Versus Single Acknowledges (DLE 0 - DLE 1) . . . . .	D-1
D.1.2.	Two-Character Responses . . . . .	D-1
D.1.3.	Negative Acknowledge (DPAK - DLE NAK) . . . . .	D-2
D.1.4.	Reply Request (DENQ - DLE ENQ) . . . . .	D-2
D.1.5.	Timers (Processor) . . . . .	D-2
D.1.6.	After the Timeout . . . . .	D-2
D.2.	EXAMPLES OF ERROR RECOVERY. . . . .	D-2
D.2.1.	Hardware Rule 5 Not Followed . . . . .	D-3
D.2.2.	Hardware Rule 5 Correctly Used With One Terminal Active on Multiplexer. . . . .	D-4

D.3.	<b>ERROR RECOVERY WITH AUXILIARY INTERFACE DEVICES</b>	D-13
D.3.1.	<b>Errors Resulting From Hits on Communications Line</b>	D-14
D.3.1.1.	Selection Poll	D-14
D.3.1.2.	Selection With Text	D-15
D.3.1.3.	Selection With Text and Print Command	D-18
D.3.1.4.	Auxiliary Interface Input Transfer	D-20
D.3.2.	<b>Errors Resulting From Error Conditions in the Devices</b>	D-23
D.3.2.1.	Selection	D-24
D.3.2.2.	Data Transfer or Operation	D-24

## INDEX

### USER COMMENT SHEET

### FIGURES

1-1.	UNISCOPE 100 Display Terminal and UNISCOPE 200 Display Terminal	1-1
1-2.	UNISCOPE Display Terminal Selections	1-3
2-1.	UNISCOPE Display Terminal System Configurations	2-2
2-2.	UNISCOPE Display Terminal, Typical System Configuration	2-3
2-3.	UNISCOPE Display Terminal Keyboard for Units With Full Protected Format Control	2-5
2-4.	UNISCOPE Terminal Code, Based Upon ASCII	2-6
2-5.	Direct Connection Configurations	2-14
2-6.	Remote Synchronous Configurations	2-15
2-7.	Remote Asynchronous Configurations	2-16
2-8.	Remote Synchronous Configurations (IBM 360 or 370)	2-17
2-9.	Remote Asynchronous Configurations (IBM 360 or 370)	2-18
2-10.	Message Format	2-19
2-11.	Tape Cassette	2-31
2-12.	Tape Format	2-34
3-1.	Processor-to-Terminal Message Formats	3-3
3-2.	Terminal-to-Processor Response Message Formats	3-5
3-3.	Example of Address Code Assignments for Single-Station Configurations	3-11
3-4.	Example of Address Code Assignments for a Configuration of a Combination of Terminals and Multiplexers	3-12
3-5.	Example of Address Code Assignments for a Multiplexer Network	3-12
3-6.	Example of Address Code Assignments With One Multiplexer Divided Into More Than One Logical Multiplexer Function	3-13
3-7.	Example of Address Code Assignments With Primary and Cascaded Multiplexers, One Primary/Cascaded Multiplexer Combination Being Divided Into More Than One Logical Multiplexer Function	3-13
3-8.	Text Format, Processor to UNISCOPE Terminal	3-14
3-9.	Text Format, UNISCOPE Terminal to Processor	3-14
3-10.	Cursor Addressing for Screen Display	3-15
3-11.	Data Format, Processor to UNISCOPE Terminal	3-17
3-12.	Data Format, UNISCOPE Terminal to Processor	3-18
3-13.	Roll-Down Function	3-19
3-14.	Roll-Up Function	3-20

3-15.	Character Structure, Synchronous Transmission . . . . .	3-22
3-16.	Character Structure, Asynchronous Transmission . . . . .	3-22
3-17.	Example of Protected Information Transmitted From Processor to Terminal . . . . .	3-24
3-18.	Example of Protected Information Displayed on Screen . . . . .	3-24
3-19.	Example of Unprotected Information Added to Protected Format Display . . . . .	3-24
3-20.	Example of Unprotected Information Transmission From Terminal to Processor . . . . .	3-25
A-1.	Responses by Terminal, Either Single Station or on a Multiplexer, but Without an Auxiliary Interface on the Single Station or Anywhere on the Multiplexer . . . . .	A-1
A-2.	Responses by Terminal on a Multiplexer With no Auxiliary Interface on the Responding Terminal but With an Auxiliary Interface on Some Other Terminal on the Same Multiplexer . . . . .	A-2
A-3.	Responses by Terminal With an Auxiliary Interface but not on a Multiplexer . . . . .	A-2
A-4.	Responses by Terminal With an Auxiliary Interface and on a Multiplexer . . . . .	A-3
B-1.	UNISCOPE Terminal Traffic Flow Diagram . . . . .	B-3

**TABLES**

2-1.	Control Characters Not Placed in UNISCOPE Terminal Storage . . . . .	2-11
2-2.	Keyboard Functions and Corresponding Transmitted 7-Bit Octal Codes . . . . .	2-21
2-3.	ASCII/Octal Code Conversions . . . . .	2-23
2-4.	Status Codes . . . . .	2-25
2-5.	Tape Cassette Interchangeability Conditions . . . . .	2-33
3-1.	Communications Control Characters . . . . .	3-2
3-2.	Interface Timing . . . . .	3-8
3-3.	Control Codes and Control Code Sequences . . . . .	3-9
3-4.	BCC Parity Character Coding . . . . .	3-23
3-5.	Control Characters and Corresponding Octal Codes Transmitted . . . . .	3-25
4-1.	Tape Cassette System Status Codes . . . . .	4-2

# 1. Introduction

## 1.1. GENERAL

The UNISCOPE Display Terminal (Figure 1-1) is a 2-way, remote terminal device which makes it possible to hold interactive, direct data communications with a central processor. Each UNISCOPE terminal is keyboard operated and uses a cathode-ray tube (CRT) for message display. The CRT (screen) displays messages sent from the processor and allows messages to be composed and edited before they are transmitted to the processor. The UNISCOPE 100 terminal contains a storage capacity of 960 or 1024 seven-bit characters; the UNISCOPE 200 terminal contains a storage capacity of 1536 or 1920 seven-bit characters.

An optional printer connected to the UNISCOPE terminal by an auxiliary interface channel enables a hard-copy printout of terminal information as keyed in by the operator or as sent from the processor, or both. An optional tape cassette system provides a local high-volume storage medium for the terminal.

UNISCOPE terminals can communicate over any suitable communications system with any central processor installation that complies with the electrical and procedural discipline of the terminal. Multiple UNISCOPE terminals can be connected to a single communications line interface point by means of the SPERRY UNIVAC Terminal Multiplexer. The UNISCOPE terminal can also be multidropped on a communications line and can be intermixed with SPERRY UNIVAC DCT 1000 Data Communications Terminals.

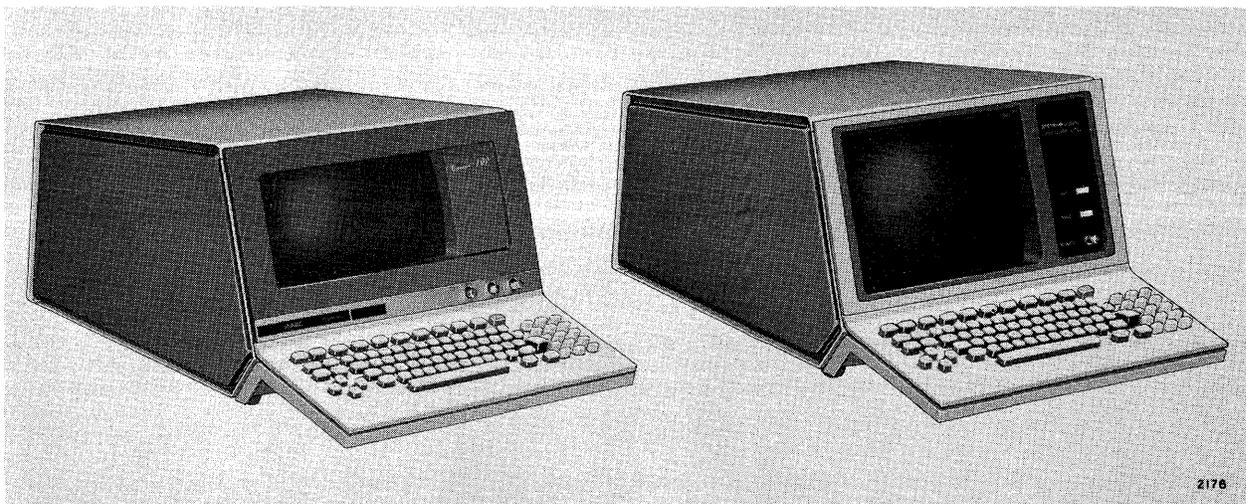


Figure 1-1. UNISCOPE 100 Display Terminal (left) and UNISCOPE 200 Display Terminal (right)

## 1.2. USER DOCUMENTS

This manual contains all information pertinent to programming of the UNISCOPE Display Terminal. Information pertaining to programming of the SPERRY UNIVAC Model 610 Tape Cassette System, used as an auxiliary device with a UNISCOPE terminal, is also contained in this manual. Information pertaining to the SPERRY UNIVAC Communications Output Printer as an auxiliary printing device for the UNISCOPE terminal is given in the *SPERRY UNIVAC Communications Output Printer Functional Description, UP-7939* (current version).

The information in this manual is based on the assumption that the user is acquainted with the current versions of the *UNISCOPE Display Terminal Concept and Applications, UP-8155*; the *UNISCOPE Display Terminal Operator Reference, UP-7788*; and the *UNISCOPE Display Terminal Operator's Guide, UP-8147*. It is not within the scope of this manual to provide information pertaining to operation and maintenance of the UNISCOPE terminal or the tape cassette system.

Information pertaining to use of the SPERRY UNIVAC Direct Connection Module is given in the *SPERRY UNIVAC Direct Connection Module Functional Description, UP-7932* (current version).

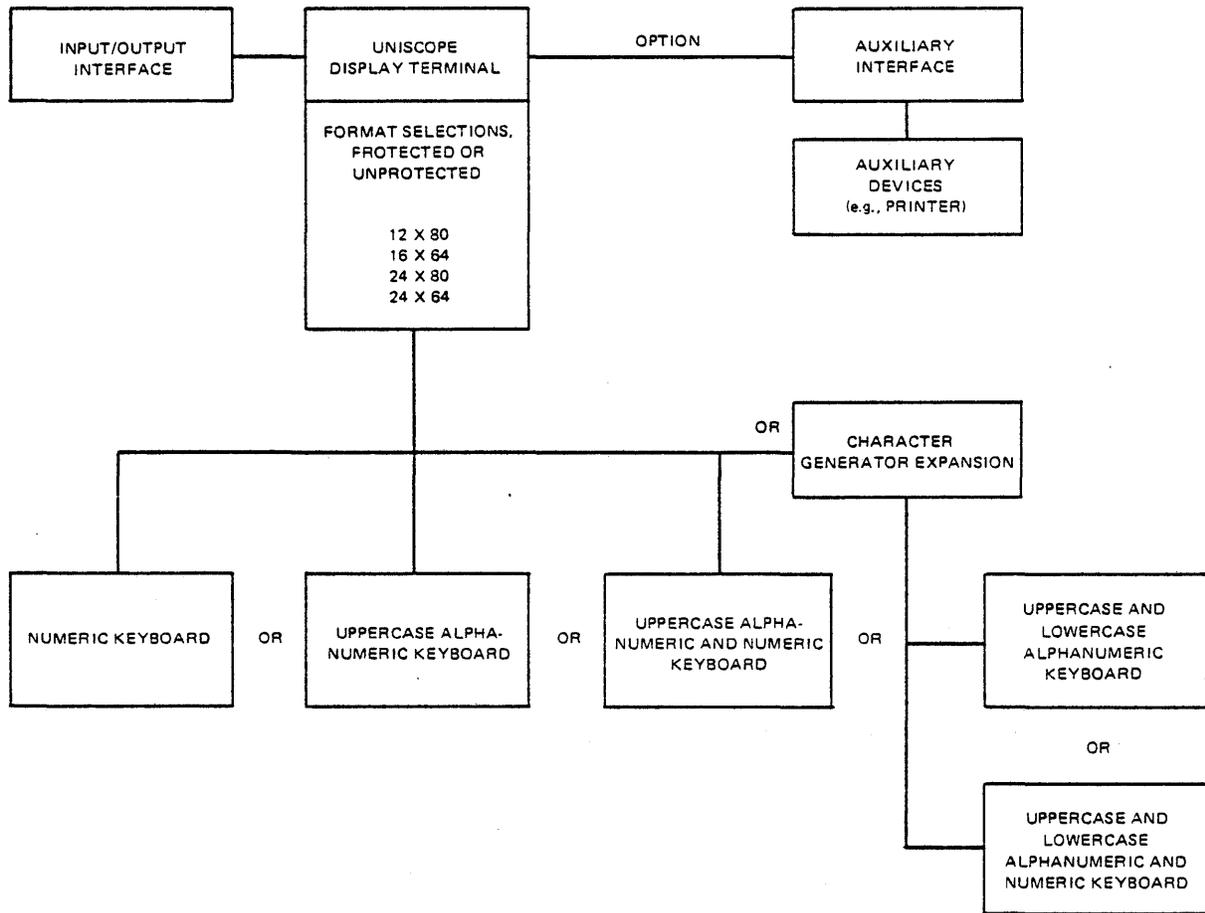
Information pertaining to use of the SPERRY UNIVAC Terminal Multiplexer is given in the *SPERRY UNIVAC Terminal Multiplexer Functional Description, UP-7916* (current version).

The user is also expected to have access to the appropriate manuals on the system in which the UNISCOPE terminal is being used.

## 1.3. TERMINAL CONFIGURATIONS

The UNISCOPE terminal may be used as a receive-only terminal or as a receive-transmit terminal with a variety of keyboards (Figure 1-2). The generator expansion feature increases the capacity of the basic character generator from 64 to 96 characters. (Use of the generator expansion feature is required with keyboards having lowercase capability.)

Auxiliary device configurations are discussed in paragraphs 2.6 and 2.8.2 of this manual and in the *SPERRY UNIVAC Communications Output Printer Functional Description, UP-7939*.



2242

Figure 1-2. UNISCOPE Display Terminal Selections



## 2. Functional Description

### 2.1. GENERAL CHARACTERISTICS OF UNISCOPE DISPLAY TERMINAL

Two types of UNISCOPE Display Terminals are available. The basic terminal has a 64-character generator capable of displaying 960 or 1024 characters on the UNISCOPE 100 terminal, or 1536 or 1920 characters on the UNISCOPE 200 terminal. A generator expansion feature expands the 64-character generator (uppercase, numeric, and symbolic characters) to a 96-character generator (uppercase, lowercase, numeric, and symbolic characters).

All models of the UNISCOPE terminal conduct communication with the central processor using ASCII (American Standard Code for Information Interchange) characters.

To operate more than one UNISCOPE terminal with a processor at a single interface point, the SPERRY UNIVAC Terminal Multiplexer may be used. This arrangement allows for up to 16 terminals on one system interface, or, if more than 16 terminals are to be on the interface, another terminal multiplexer is cascaded from the first one, providing for 31 terminals. (Only one level of cascading is allowable.) This level of cascading can be continued until a maximum of 256 terminals are connected at one system interface point. The actual number of terminals that can be accommodated at one system interface point depends on several factors, such as the expected amount and length of traffic from individual terminals and the software handler techniques employed. UNISCOPE terminal communications control procedures are not affected by the presence of SPERRY UNIVAC DCT 1000 Data Communications Terminals; both terminal types obey the same procedures. Refer to 2.3 for additional information on use of the terminal multiplexer.

The UNISCOPE terminal and terminal multiplexer interface features are provided as options for direct connection to a SPERRY UNIVAC Communications Terminal Module Controller (CTMC) or SPERRY UNIVAC Data Communications Subsystem (DCS) and for connection to synchronous and asynchronous modems or modem substitutes such as the SPERRY UNIVAC Direct Connection Module (DCM). Optional features for the UNISCOPE terminals, terminal multiplexer, and associated interfaces are described in 2.4.

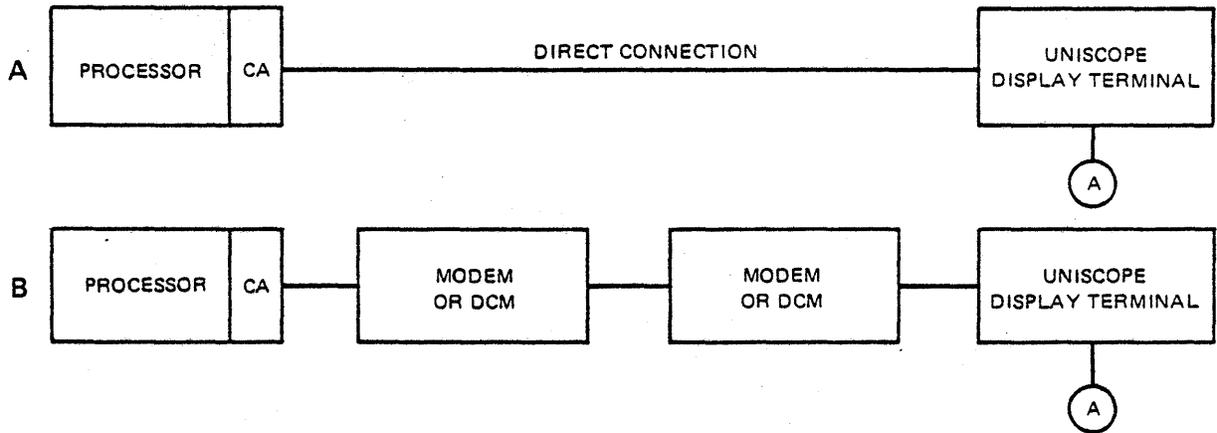
Messages from the processor are routed to a UNISCOPE terminal according to the message code; the messages are placed in the terminal storage and are displayed on the screen. The optional printer or tape cassette system can be accessed by way of the auxiliary interface for printouts or storage of the display.

Messages from the UNISCOPE terminal keyboard are displayed on the screen before being transmitted to the processor. The optional printer or tape cassette system can be accessed by way of the auxiliary interface for message printout or retrieval from storage.

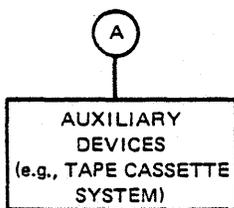
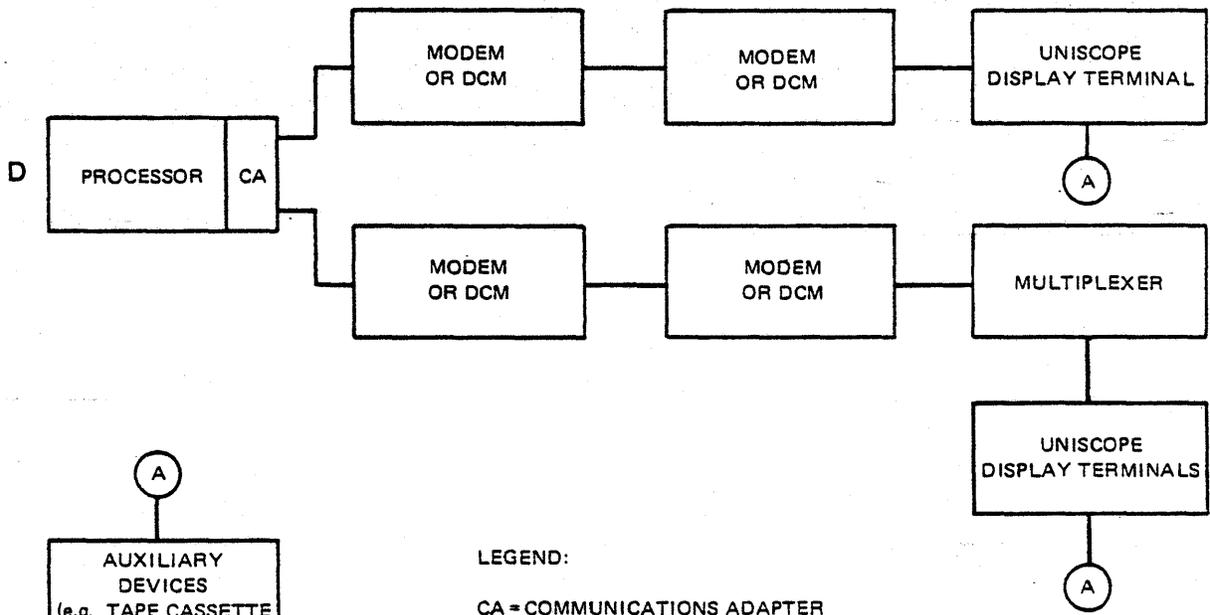
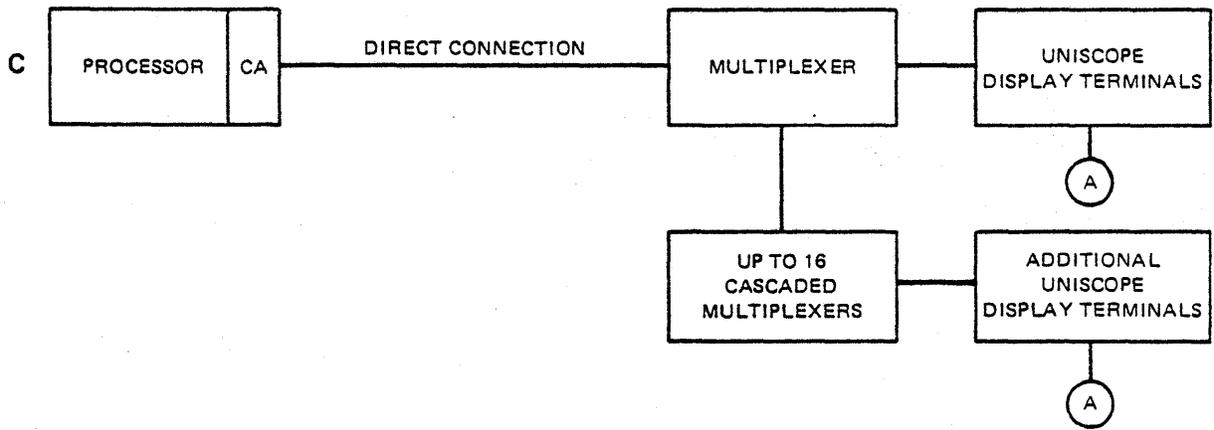
To promote data integrity, both the UNISCOPE terminal and the processor send acknowledgments for suitable messages that are correctly received. The terminal automatically returns an acknowledgment to suitable processor messages in the poll response. The processor acknowledges suitable terminal messages in the next poll to the terminal.

Figure 2-1 shows the various system configurations in which the UNISCOPE terminal can be used. Figure 2-2 shows a typical configuration for a UNISCOPE Display Terminal system.

SINGLE-STATION CONFIGURATIONS



MULTISTATION CONFIGURATIONS



LEGEND:

CA = COMMUNICATIONS ADAPTER

Figure 2-1. UNISCOPE Display Terminal Systems Configuration

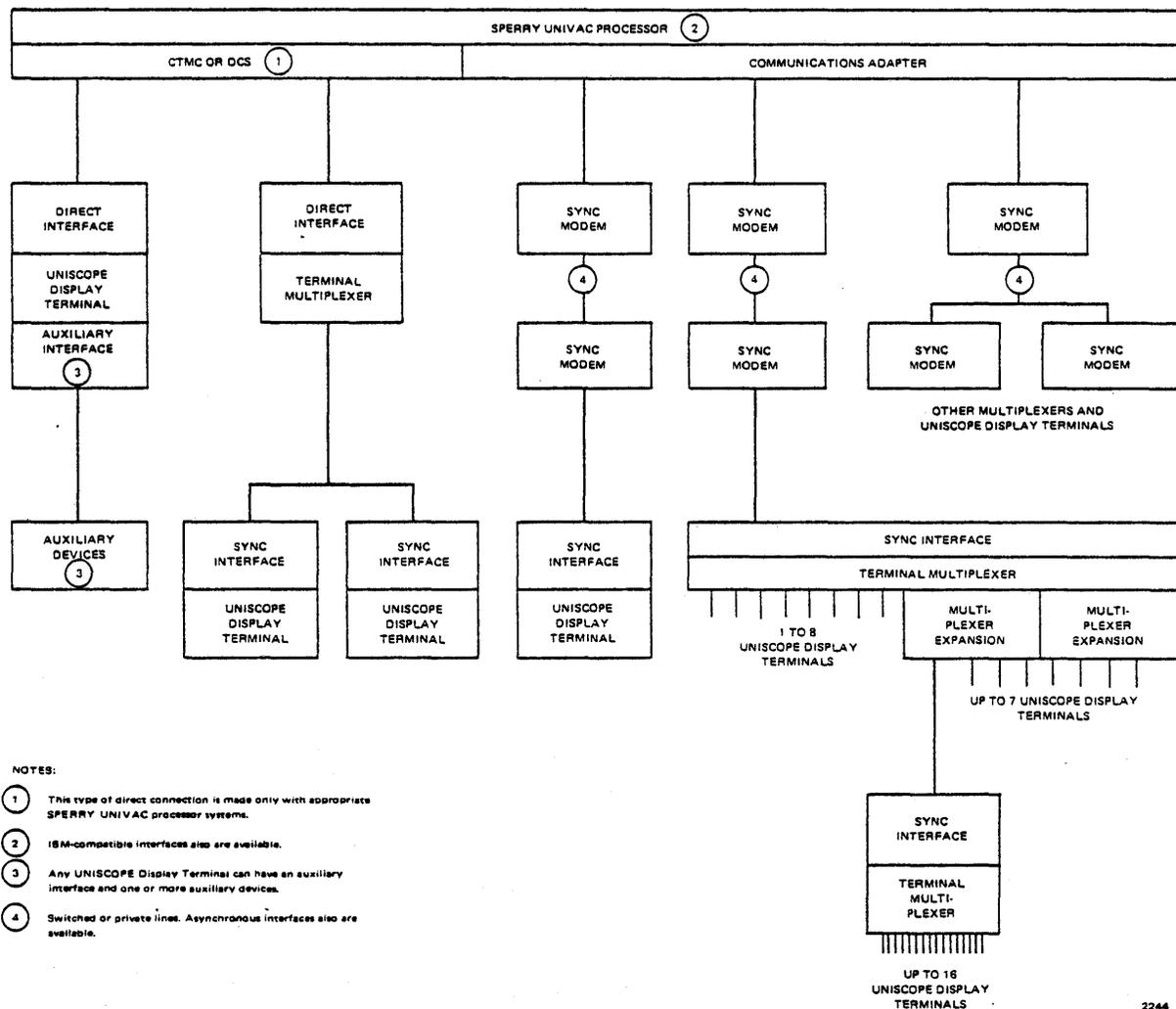


Figure 2-2. UNISCOPE Display Terminal, Typical System Configuration

## 2.2. OPERATION OF UNISCOPE DISPLAY TERMINAL

The UNISCOPE terminal is a self-contained unit consisting of a display screen, keyboard, character generator, storage, message buffering, and control logic. Data may be entered at any character location in the storage and on the screen by means of the keyboard, the processor, or the auxiliary interface.

A protected format feature may be selected for the UNISCOPE terminal. This feature enables protection of data fields that are established by means of processor programs. The processor program establishes a stencil or file overlay which defines the protected and unprotected data fields of individual files. The logic of the protected format prevents an operator from overwriting or modifying these protected fields in normal operation.

### 2.2.1. Cathode-Ray Tube Display Screen

The cathode-ray tube (CRT) screen measures 10 inches by 5 inches on the UNISCOPE 100 terminal and 10 inches by 7 inches on the UNISCOPE 200 terminal. The screen is suitable for displaying up to the full display capability of alphanumeric characters in readable size.

### 2.2.2. Cursor

The cursor (  $\square$  for UNISCOPE 100 terminal,  $\blacksquare$  for UNISCOPE 200 terminal) is a unique character that is constantly displayed on the CRT (except for a few moments when transmitting or transferring via the auxiliary interface) and marks the position that will be occupied by the next character entered into storage (and on the screen). It also marks the last character of text when data is transmitted to the processor. The first character of text transmitted is marked by the last start-of-entry (SOE) character, or by the home position if no SOE is used, before the cursor.

The cursor is nondestructive; that is, when it is placed over a character (displayable or nondisplayable), it does not replace that character in storage. Instead, the cursor and the character over which it is placed alternately blink. The cursor appears steadily over a space.

The cursor can be positioned under control of the operator, the processor, or the auxiliary interface. As data is generated, the cursor advances one space at a time as a character is entered in the space that the cursor is occupying. Eleven control keys (2.2.3.4) provide a large degree of cursor manipulation. By repositioning the cursor, the operator can alter the message by overtyping. When used with the editing keys (2.2.3.5), the cursor indicates the position for inserting data in text or deleting data from text. When the cursor reaches the eighth character position from the end of a line, or reaches the bottom line of the display, an alarm (beep) is sounded.

#### NOTE:

*Refer to 4.2.2 and 4.3.2 for information related to cursor positioning when transferring data to or from the tape cassette system.*

### 2.2.3. Keyboard Operation

Messages, consisting of alphanumeric characters, punctuation, special characters, and spaces, are composed by means of the keyboard. As a character is entered from the keyboard, it is stored in a buffer at the current cursor position, the cursor is advanced one position, and the entire buffer is displayed on the screen (2.2.2). Detailed operating instructions are provided in the *UNISCOPE Display Terminal Operator Reference, UP-7788* (current version) or *UNISCOPE Display Terminal Operator's Guide, UP-8147* (current version).

The operator can transmit any portion of the displayed material as a message by placing the cursor at the desired starting point of the message, pressing the start-of entry (SOE  $\triangleright$ ) key, moving the cursor to the end of the desired message, and pressing the TRANSMIT key. The keyboard is then disabled until the next processor text message or acknowledgment (depending on the terminal strapping) is received at the UNISCOPE terminal. When multiple SOE characters are on the screen, transmission starts at the last SOE before the cursor. If no SOE characters are on the screen, transmission begins with the home position (first position in the upper left corner of the screen).

The keyboards are divided into several sections (Figure 2-3) according to function:

1. Alphanumeric/symbolic section
2. Numeric section
3. Message control section
4. Cursor control section
5. Editing section

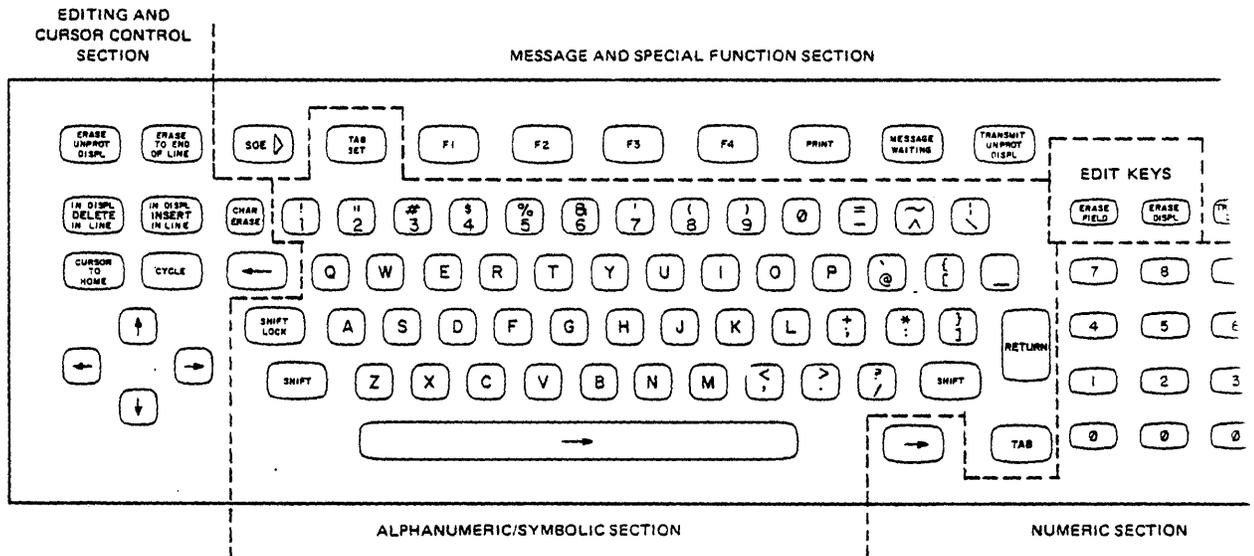


Figure 2-3. UNISCOPE Display Terminal Keyboard for Units With Full Protected Format Control

6. Special function section
7. Protected format section

Several keyboard configurations are available according to the variations of the alphanumeric/symbolic and numeric keyboard sections. All of the keyboards include the message control, cursor control, and editing sections.

### 2.2.3.1. Alphanumeric/Symbolic Section

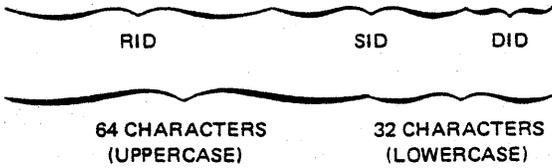
The keys of this section (Figure 2-3) are used to compose the message which is displayed on the screen prior to being transmitted to the processor. Both uppercase and uppercase/lowercase options are available for the operator to key in the ASCII graphic character set shown in Figure 2-4.

The uppercase option uses columns 2 through 5 (ASCII), and the shift key is used for shifting the data keys in the alphanumeric/symbolic section to uppercase.

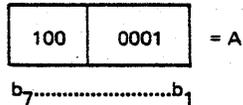
### 2.2.3.2. Numeric Section

The numeric section (Figure 2-3) provides a fast and convenient way to enter numerals into the display for transmission to the processor. The resulting codes are identical to those of the numerals and plus, minus, and decimal point (period) symbols of the alphanumeric/symbolic section. (The plus, minus, and decimal point are not included in the numeric section on keyboards having the protected format control.) A space key, separate from the alphanumeric/symbolic space bar and close to the numeric key section, is included for convenience. This section is not affected by the shift key; numerals are produced in either the uppercase or lowercase shift level.

		CONTROL CHARACTERS			GRAPHIC CHARACTERS					
		COL.	0	1	2	3	4	5	6	7
ROW	BITS	000	001	010	011	100	101	110	111	
0	0000	NUL	DLE	SP	0	@	P	\	P	
1	0001	SOH	DC1	!	1	A	Q	a	q	
2	0010	STX	DC2	"	2	B	R	b	r	
3	0011	ETX	DC3	#	3	C	S	c	s	
4	0100	EOT	DC4	\$	4	D	T	d	t	
5	0101	ENQ	NAK	%	5	E	U	e	u	
6	0110	ACK	SYN	&	6	F	V	f	v	
7	0111	BEL	ETB	'	7	G	W	g	w	
8	1000	BS	CAN	(	8	H	X	h	x	
9	1001	HT	EM	)	9	I	Y	i	y	
10	1010	LF	SUB	*	:	J	Z	j	z	
11	1011	VT	ESC	+	;	K	[	k	{	
12	1100	FF	FS	,	<	L	\	l		
13	1101	.CR	GS	-	=	M	]	m	}	
14	1110	SO	RS	.	>	N	^	n	~	
15	1111	SI	US	/	?	O	-	o	//	



Example:



NOTES:

1. The specific address identifiers (RID, SID, and DID) are chosen from the columns as indicated.
2. The general address identifiers (GID) are designated SP for RID, P for SID, and p for DID.
3. The // symbol (column 7) is the symbol for the ASCII delete character (DEL).

2178

Figure 2-4. UNISCOPE Terminal Code, Based Upon ASCII

### 2.2.3.3. Message Control Section

The following keys comprise the message control section (Figure 2-3) to aid in the manipulation of data between the terminal station and the processor or auxiliary devices.

- SOE  
Start of entry
- PRINT  
When pressed, enables auxiliary interface data transfers.
- MESSAGE WAIT  
When pressed, sends a special message which can be interpreted by software; for example, it requests an unsolicited processor message to appear on the screen.
- TRANSMIT\*  
When pressed, results in transmission of a message to the processor following polling.

### 2.2.3.4. Cursor Control Section

The following keys in the cursor control section (Figure 2-3), when pressed, control movement of the cursor:

-  Scan right
-  Space bar
-  Space key (numeric section)
-  Scan left
-  Backspace
-  Scan up
-  Scan down
- RETURN      Cursor return
- CURSOR  
TO  
HOME
- TAB  
SET
- TAB

*\*Throughout this manual, this keycap name will be used to simplify reference; it will refer to the TRANSMIT keycap on unprotected units and to the TRANSMIT UNPROT DISPL keycap used on the protected format units.*

### 2.2.3.5. Editing Control Section

The following keys in the upper left corner of the keyboard (Figure 2-3) are used for making changes to the displayed data before it is transmitted to the processor (or to the auxiliary interface).

- CHAR  
ERASE
- ERASE  
TO END  
OF LINE
- ERASE TO  
END OF  
DISPL\*
- IN DISPL  
INSERT  
IN LINE
- IN DISPL  
DELETE  
IN LINE
- CYCLE

### 2.2.3.6. Special Function Section

The special function keys, located in the center of the top row of keys and labeled F1, F2, F3, and F4, are used to generate four unique characters, one per key, that are used as flags or indicators to the program to perform a variety of tasks. The operator can generate and transmit a special function message by simply pressing the desired key. This message will contain the start-of-heading, address characters, special function character, end-of-text character, and block check character. It may also contain an acknowledge or busy signal. The special function codes are ASCII 7 (key F1), G (key F2), W (key F3), and g (key F4). These keys are never locked by the keyboard lock function. They are, however, locked with respect to each other; pressing one locks the others. An acknowledge to the special function message unlocks the special function keys.

### 2.2.3.7. Protected Format Section

The protected format function prevents an operator from overwriting, modifying, or erasing protected fields in normal operation. The following keys comprise the protected format section. Two of the keys provide protected format use without giving the operator control of the protected format data; three of the keys provide the operator with protected format control (the ability to change protected format data).

- ERASE  
UNPROT  
DISPL\*\*

*\*Throughout this manual, this keycap name will be used to simplify reference; it will refer to the ERASE TO END OF DISPL keycap used on unprotected format units and to the ERASE UNPROT DISPL keycap used on the protected format units.*

*\*\*Throughout this manual, references to this key will be included in the unprotected format label, ERASE TO END OF DISPL.*

- TRANSMIT  
UNPROT  
DISPL \*
- ERASE  
FIELD
- ERASE  
DISPL
- TRANSMIT  
DISPL

#### 2.2.4. Control and Indicator Panel

The following controls and indicators are used to control the keyboard, screen, and power and to indicate message status. Except for the enable/disable switch, they are located on the face of the terminal.

- MESSAGE  
WAIT  
indicator
- WAIT  
switch and indicator

The indicator lights when the keyboard is locked. This switch acts as a master clear control, terminating any function currently in process. Incorrect use of the control can result in the loss of polls, termination of poll responses from the UNISCOPE terminal in midtransmission, or destruction of text to the terminal. On the UNISCOPE 100 terminal, the switch and indicator are combined into one control but the function is the same as described.

- MESSAGE  
INCOMPLETE  
indicator
- Audible alarm
- INTENSITY  
control
- POWER  
switch and indicator

Applies primary power and a master clear to the terminal unit when pressed; lights to indicate that power is applied.

- POLL  
indicator

Installed only on UNISCOPE 200 terminals, this indicator lights for approximately 1/2 second upon correct completion of a poll or message to the unit.

- Enable/disable switch

### 2.2.5. Message Buffering

The UNISCOPE terminal contains a buffer area for storing messages. Text characters supplied by the processor are checked for correct parity and transferred to the display buffer, where the information is stored for display on the screen. Readout from the display storage is nondestructive so that the display may be regenerated at the 60-cycle rate (50-cycle rate, if applicable).

### 2.2.6. Keyboard Lock Function

The keyboard is locked whenever the operator presses the TRANSMIT, PRINT, or TRANSMIT DISPL key or when a text message is being received. The keyboard is also locked when the enable/disable switch is in the disable position. The keyboard lock function never locks the MESSAGE WAITING key or the special function keys.

Correct receipt of a text message results in automatic unlocking of the keyboard, unless the processor sends a keyboard lock code DC4 or ESC DC4. Following a TRANSMIT condition, the keyboard is unlocked by either an acknowledgment or a text message from the processor, depending on the terminal strapping. Completion of a PRINT function results in automatic unlocking of the keyboard. Incomplete functions will cause the keyboard to remain locked. Manual use of the WAIT switch unlocks the keyboard, since this control acts as a master clear control. Any function currently in process when this manual operation is performed is terminated. Use of the control can result in loss of polls, termination of poll responses from the UNISCOPE terminal or of passed responses from other terminals on multipoint networks, or the destruction of text to the terminal.

### 2.2.7. Processor-Controlled Operation

The processor transmits messages to the UNISCOPE terminal by using the same code characters as those on the keyboard plus a group of special control functions. Detailed editing and device control code information is given in 3.6.5.

Most messages transmitted from the processor to a UNISCOPE terminal are in response to an operator enquiry, and they can be transmitted without conflicting with operator activity. For an unsolicited message, the processor can transmit a code to the terminal which lights the MESSAGE WAITING indicator and sounds the audible alarm to advise the operator of a waiting message. The operator can subsequently request the message by pressing the MESSAGE WAITING key. This key is not locked with the rest of the keyboard after the TRANSMIT key is pressed. The processor also has the unconditional ability to force the message to be displayed and thus to override any other operation (3.4.1.1). The optional printer can be controlled by the processor as well as by the operator, and hard copy of the processor message, as well as keyed data, can be obtained. Unsolicited messages can be relayed by the processor from one remote terminal to another remote terminal on the same communications network.

Data from the processor is displayed by transmitting the message directly into the buffer storage in forward sequence, starting at the screen address specified at the beginning of the message; hence, the processor designates the location at which a character is to be displayed (cursor control). All data characters are stored. Certain control characters which are contained in the message are also stored. Other control characters (such as cursor return or tab) are not stored but rather initiate a designated function when they are received. Table 2-1 lists those control characters that are not placed in storage. No control characters supplied via the communications interface are stored unless they occur within the message (between STX and ETX).

The following paragraphs explain the processor-controlled display on the terminal screen.

Table 2-1. Control Characters Not Placed in UNISCOPE Terminal Storage

Communications Interface				Auxiliary Interface
Before STX or After ETX		Following STX and Before ETX		
Synchronous	Asynchronous	Synchronous	Asynchronous	
None stored	None stored	NUL ETX HT CR SO* SI* DC1 DC2 DC4** SYN ESC DLE	NUL ETX HT CR SO* SI* DC1 DC2 DC4**  ESC DLE	NUL ETX HT CR SO* SI* DC1 DC2 DC4**  ESC

\*Not stored on protected format units.

\*\*Stored on UNISCOPE 100 terminals with serial numbers 6000 to 14,000.

### 2.2.7.1. Request Retransmission

If the terminal sends a reply request to the processor, the processor may send a poll with acknowledgment or a retransmission request. If the processor has not received a correct terminal message and wants the message retransmitted, it sends a retransmission request. If locked, the keyboard will remain locked.

### 2.2.7.2. Field Blinking

Data fields in a processor message may be bracketed by displayed blink marker characters (▢ and ▣). During alternate half-second periods, the blink marker characters are not refreshed, causing them to blink. The blinking is stopped under processor control by deleting the marker characters. The operator can also eliminate the blinking markers by writing over them or by erasing them. Multiple sets of blink markers can be used simultaneously; the only limit is the size of the display storage.

### 2.2.7.3. Line Insert

By control codes in a processor message, a blank line can be created anywhere in the display. The data previously in that line location, and in all subsequent lines, is shifted downward one line position. Any data in the bottom line of the screen is removed from the display. The processor-positioned cursor will be in column 1 of the line to be made blank for line insertion; and when the data to be entered is transmitted, the cursor will progress across the blank line, one character at a time, as the new line of data is corrected. With protected format, the line insert function is the same.

### 2.2.7.4. Line Delete

By control codes in a processor message, the processor can delete any selected line; all data below the line then shifts upward one line position. This creates a blank line at the bottom of the screen. The processor-positioned cursor will be in column 1 of the line to be deleted and will be in the same position when the remaining lines move up on the screen. If data is to be inserted in the blank line created at the bottom of the screen, the processor must reposition the cursor to the first position of that line. The cursor then progresses across the line, one character at a time, as new data is entered. With protected format, the line delete function is the same.

### 2.2.7.5. Display Rolling

The display can be made to roll upward or downward by control of the processor line insert or line delete codes to produce a moving scroll effect (3.5.3).

### 2.2.7.6. Horizontal Tab

The processor can insert tab stop set codes in any location in the display storage. When the operator presses the TAB key, the cursor moves to the character position on the right of the first tab location. With protected format, if the tab stop is in a protected field, the cursor moves to the first unprotected character location after the tab stop.

### 2.2.7.7. Cursor Return/Line Feed

On all but the last line, the cursor return/line feed function is automatic at the end of the line. At the end of the last line, the cursor moves to the home position. For shorter lines, the processor can insert in the message a return code which initiates the function but is not stored.

### 2.2.7.8. MESSAGE WAIT Indicator

Receipt of the BEL code in a specially formatted message causes the terminal to light the MESSAGE WAIT indicator and intermittently sound an alarm. When the operator presses the MESSAGE WAIT key or a function key and that message is sent, the MESSAGE WAIT indicator is extinguished.

## 2.3. TERMINAL MULTIPLEXER CONNECTIONS

Up to 16 terminals can be connected to a single interface point by means of one SPERRY UNIVAC Terminal Multiplexer. By cascading multiplexers, up to 256 terminals can be connected to one interface point. The practical limit will vary according to the expected amount of traffic, the expected length of messages, and the software handler techniques used.

The basic operational requirement of the SPERRY UNIVAC Terminal Multiplexer is to control terminal transmission sequences. Processor messages being sent to one or more terminals are not affected by the multiplexer. All terminals on the first multiplexer and on cascaded multiplexers will see all receive messages, both text and poll (unless a terminal is transmitting on a shared full-duplex line). The receive portion of the multiplexer looks like one common conductor going to all branches simultaneously. The terminals themselves determine what information they will accept. Only if the address accompanying the message is recognized by the terminal will it be able to distinguish whether the message is a poll message or a text message. The address must be either specific (applying only to one terminal) or general (applying to several terminals).

All the terminals on the first multiplexer and on each cascaded multiplexer will see either a general poll, a specific poll, or text (received data) at approximately the same time. Cable propagation time between multiplexers and terminals makes a slight difference in receive time.

The primary purpose of the multiplexer is to select, one at a time, those terminals and cascaded multiplexers (with attached terminals) that desire to send information to the central processor. This could be text, acknowledge, busy, auxiliary interface selection status, message waiting, or special-function key information. The multiplexer can pass an ACK or WABT from one branch to another within that multiplexer. It can select, on a priority basis, those terminals and cascaded multiplexers (with attached terminals) that desire to send information to the central processor. Refer to *SPERRY UNIVAC Terminal Multiplexer Functional Description, UP-7916* (current version) for detailed information on the operation of the multiplexer.

## 2.4. COMMUNICATIONS INTERFACE BETWEEN UNISCOPE TERMINAL AND PROCESSOR

To provide compatibility with the communications facilities, specific communications equipment and terminal features are required for interfacing with either half-duplex or full-duplex switched or private line voice-grade telephone networks. The equipment requirements for the remote and central sites are described in the following paragraphs.

The UNISCOPE Display Terminal can use (but is not limited to) any of the following communications interface methods in operating with a processor.

- Direct connection to a communications terminal adapter. This permits central site direct synchronous communications through a SPERRY UNIVAC Data Communications Subsystem (DCS) to the SPERRY UNIVAC 9000 Series processor or through a SPERRY UNIVAC Communications Terminal Module Controller (CTMC) to the SPERRY UNIVAC 400 Series and 1100 Series processors. See *SPERRY UNIVAC Communications Terminal Module Controller Programmer/Operator Reference, UP-7519* (current version).
- Remote connection to SPERRY UNIVAC processors over common-carrier lines through modems. Communications can be in either the synchronous or asynchronous mode.
- Remote connection to the IBM\* System/360 or System/370 processor over common-carrier lines through modems. Communications can be in either the synchronous or asynchronous mode.
- Connection through modem substitutes such as the SPERRY UNIVAC Direct Connection Module (DCM).

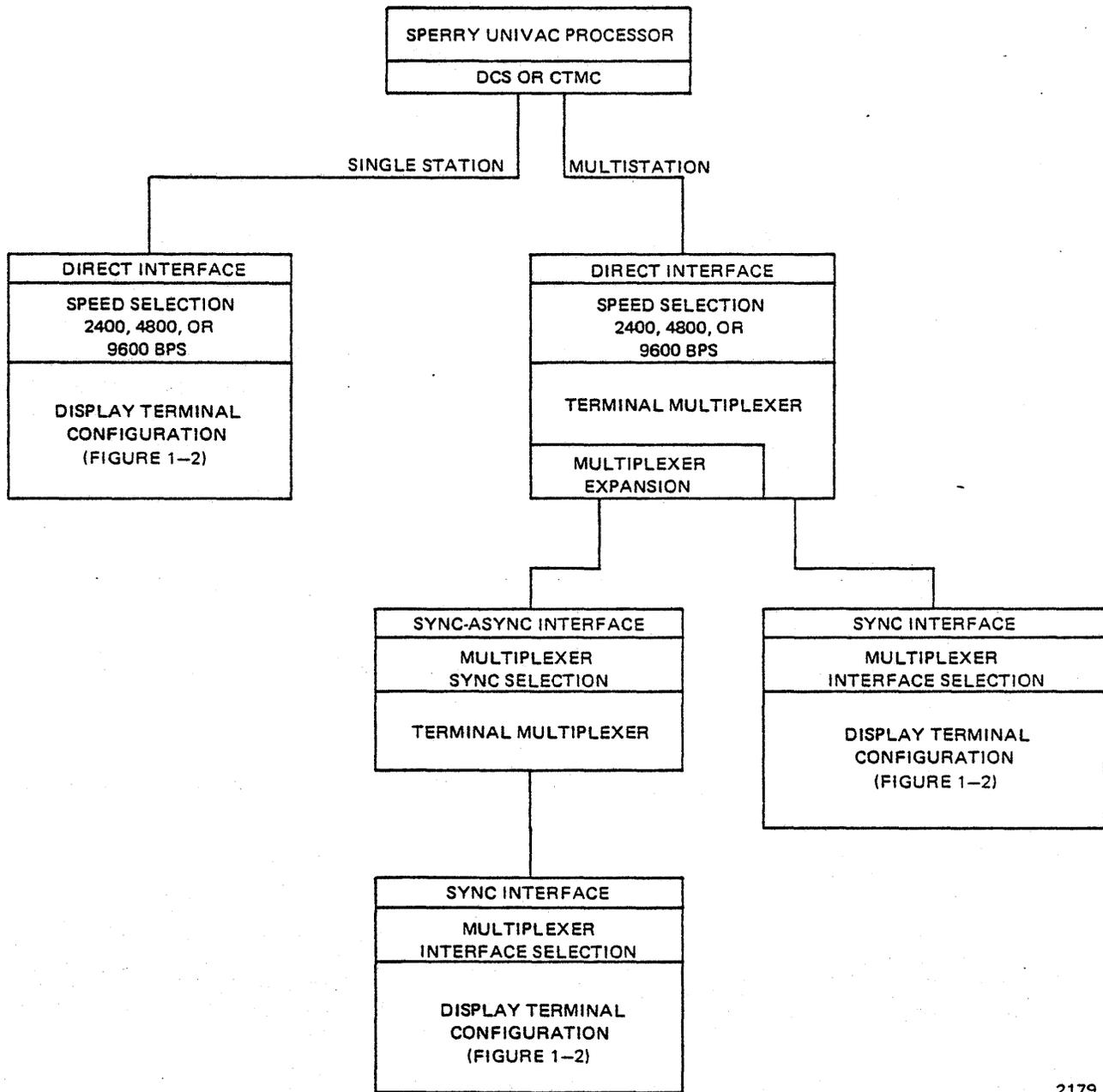
The UNISCOPE terminal can use these interface methods either in a single-station application or in multidrop system configurations as determined by the connection devices (2.3).

Remote sites require modems and synchronous or asynchronous interface features for either multiplexed terminals or single terminals. Special interfacing features are used for certain distances between the UNISCOPE terminal and the multiplexer. There are also special interfaces required between the terminal multiplexer and the modems.

### 2.4.1. Direct Connection

Central site single or multistation terminals can be connected for synchronous operation directly to a SPERRY UNIVAC 9000 Series processor through a DCS or to a SPERRY UNIVAC 400 Series or 1100 Series processor through a CTMC (Figure 2-5). Operating speeds to 9600 bits per second (bps) are available. The DCS requires communications interface feature F1002-08 in a direct-connection application. The CTMC requires the F1018-XX high-speed interface module.

\*Trademark of International Business Machines Corporation

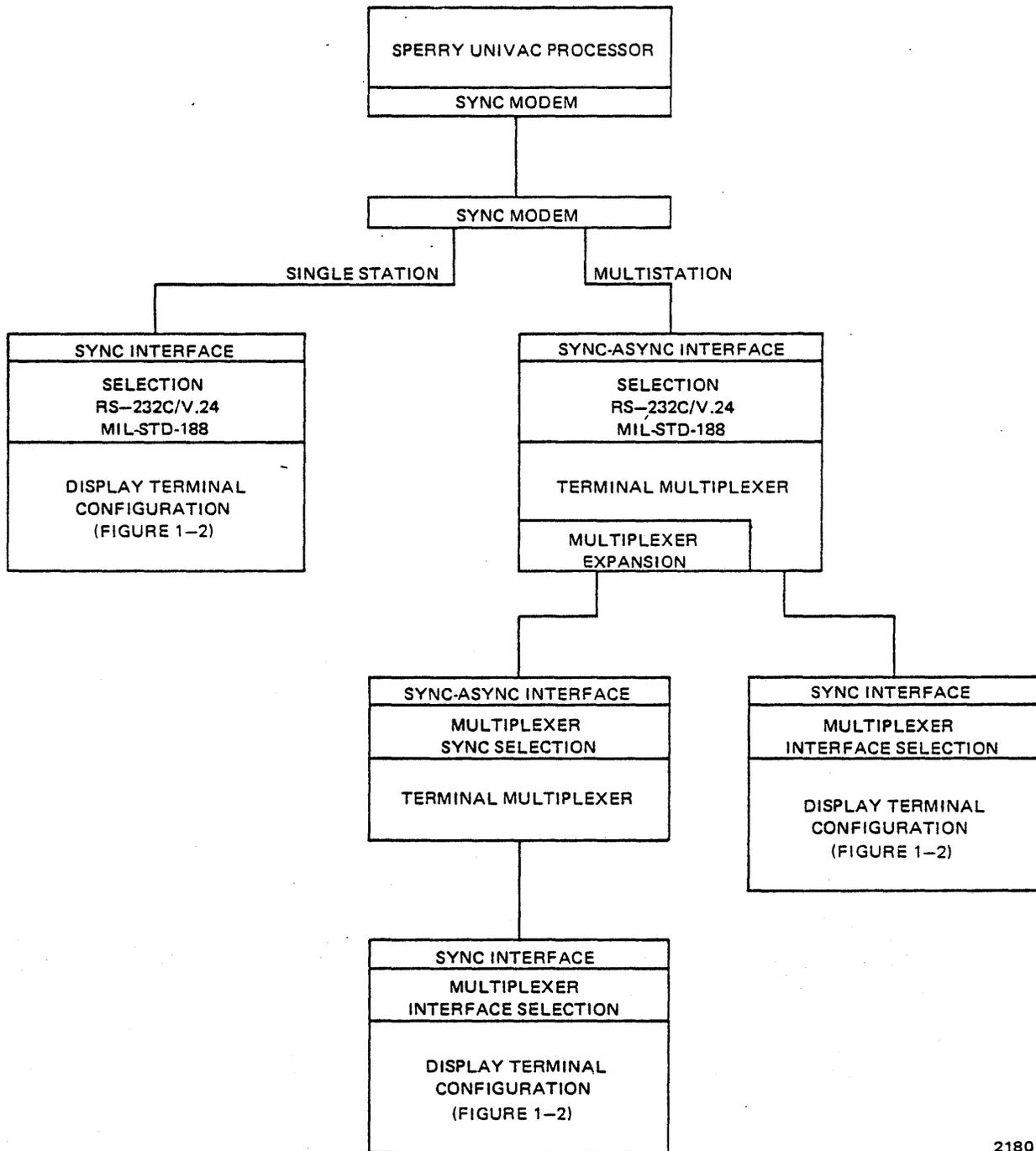


2179

Figure 2-5. Direct Connection Configurations

#### 2.4.2. Remote Synchronous Operation (SPERRY UNIVAC Processors)

The system configurations for remote single-station or multistation synchronous operation with a SPERRY UNIVAC processor (Figure 2-6) are similar to those for direct connection shown in Figure 2-5. Differences include the requirements for modems for remote operation and the operating speeds available. Operating speeds for synchronous operation are up to 9600 bps, as determined by the modems used.



2180

Figure 2-6. Remote Synchronous Configurations

### 2.4.3. Remote Asynchronous Operation (SPERRY UNIVAC Processors)

The system configurations for remote single-station and multistation asynchronous operation with a SPERRY UNIVAC processor (Figure 2-7) are similar to those for direct connection shown in Figure 2-5. Differences include the requirements for asynchronous interfaces and for modems for remote operation, and the operating speeds available. Operating speeds for asynchronous operation are 300, 600, 1200, 1600, 1800, and 2400 bps in single-station applications through modems complying with EIA Standard RS-232-C and CCITT Recommendation V.24; 1200, 1600, 1800, and 2400 bps in single-station applications through modems complying with military standard MIL-STD-188; and 300, 600, 1200, 1600, 1800, and 2400 bps for multistation configurations. (The choice of modems must reflect the specific transmission speed.)

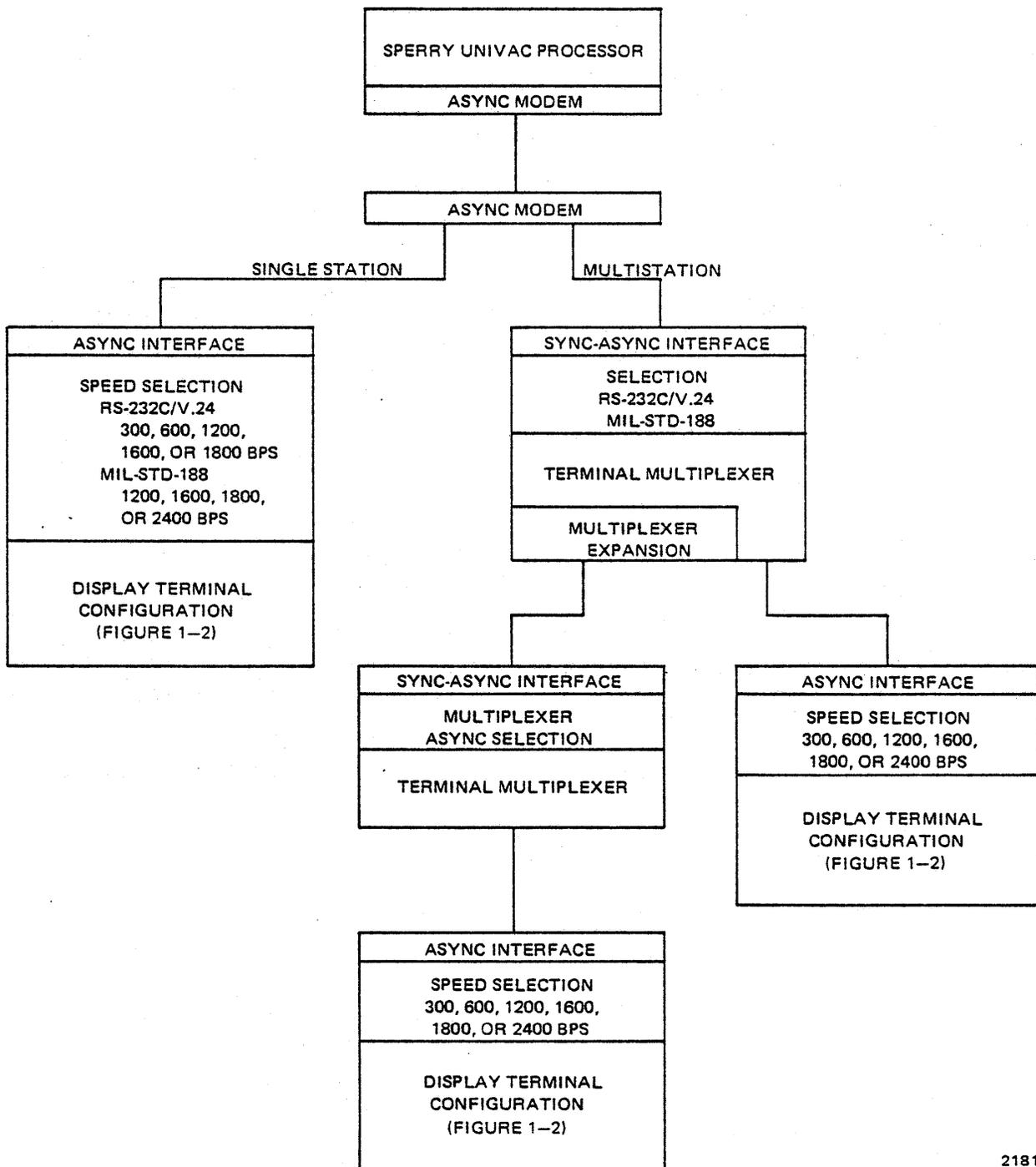


Figure 2-7. Remote Asynchronous Configurations

### 2.4.4. Remote Synchronous Operation (IBM System/360 or System/370)

The system configurations for remote single-station or multistation synchronous operation with IBM System/360 or System/370 processors (Figure 2-8) are similar to those for remote synchronous connection shown in Figure 2-6. Differences include the requirement for modems for remote operation and the operating speeds available. Operating speeds up to the terminal hardware limitations are determined by the whole system. These configurations require that the IBM System/360 or System/370 processor be equipped with either the 2701 and SDA II or the 2703 and Synchronous Base 1 communications adapters, or a suitable emulator of these devices.

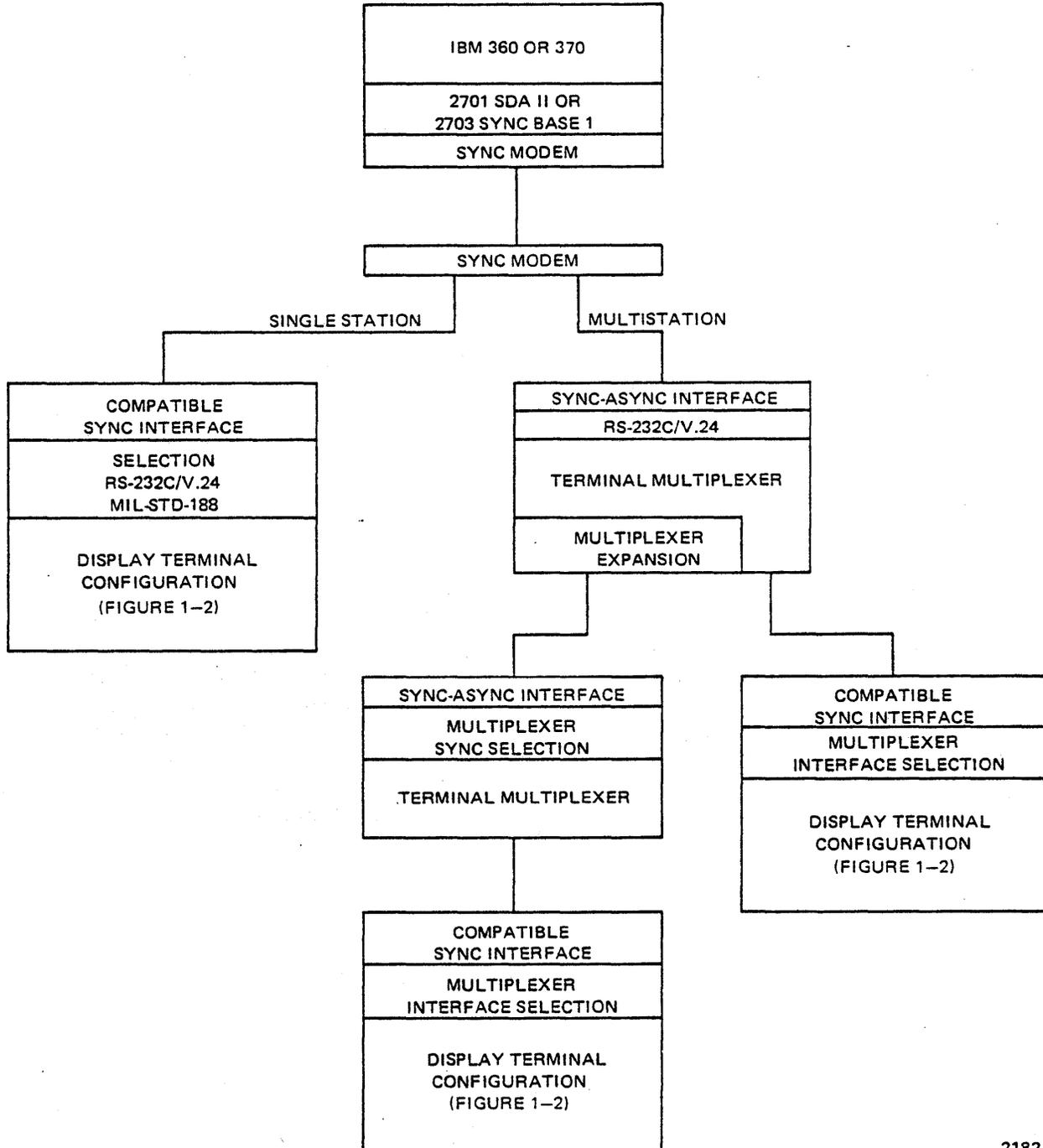


Figure 2-8. Remote Synchronous Configurations (IBM 360 or 370)

### 2.4.5. Remote Asynchronous Operation (IBM System/360 or System/370)

The system configurations for remote single-station or multistation asynchronous operation with IBM System/360 or System/370 processors (Figure 2-9) are similar to those for remote asynchronous connection shown in Figure 2-7. Differences include requirements for asynchronous interfaces and for modems for remote operation, and the operating speeds available. Operating speeds for asynchronous operations are 300, 600, 1200, 1600, 1800, and 2400 bps in single-station applications through modems complying with EIA Standard RS-232-C and CCITT Recommendation V.24; 1200, 1600, 1800, and 2400 bps in single-station applications through modems complying with military standard MIL-STD-188; and 300, 600, 1200, 1600, 1800, and 2400 bps for multistation configurations. (The choice of modems must reflect the specific transmission speed.) These configurations require that the IBM System/360 or System/370 processor be equipped with the 2701 and Terminal Adapter III communications adapters or a suitable emulator.

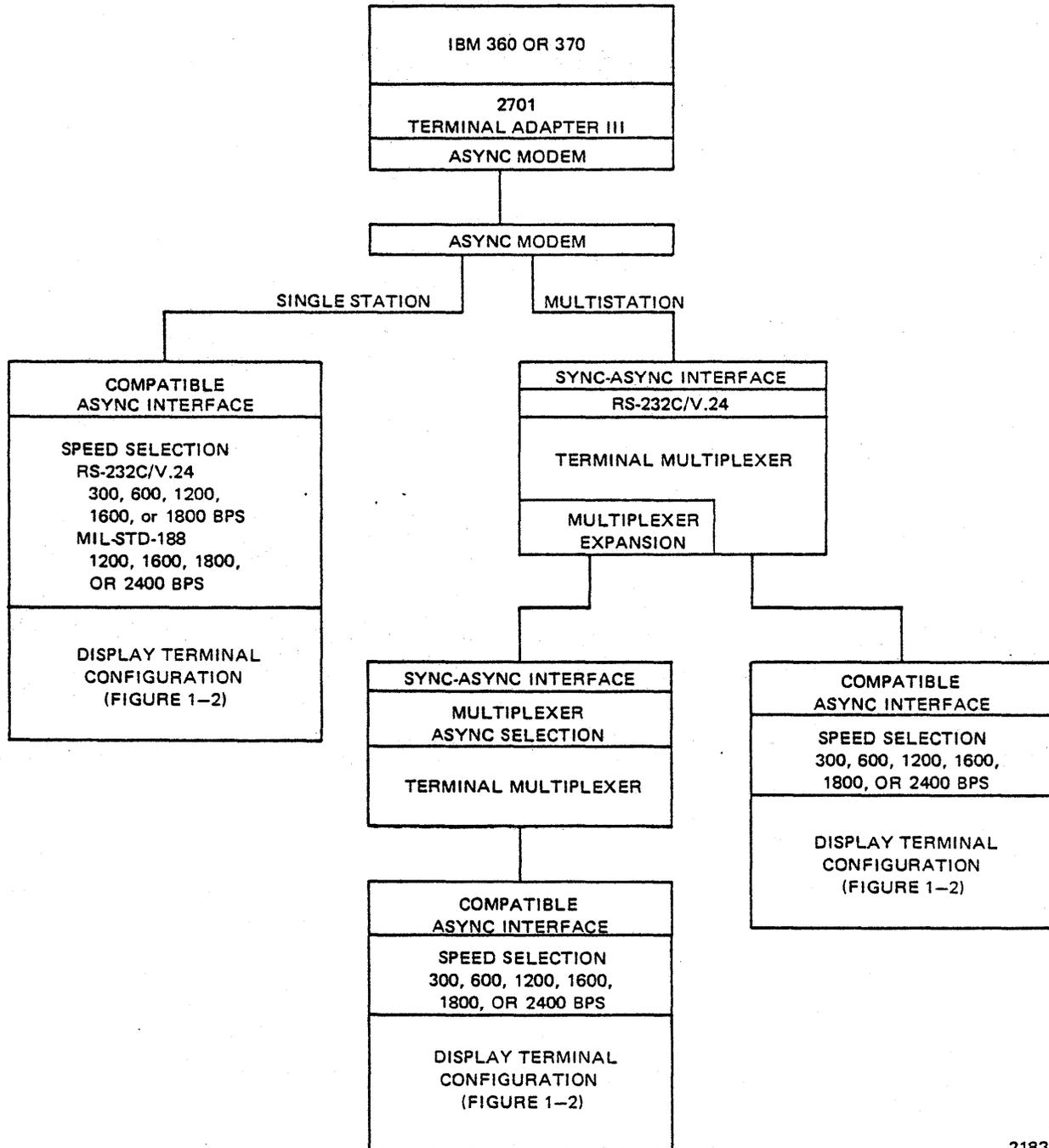


Figure 2-9. Remote Asynchronous Configurations (IBM 360 or 370)

## 2.5. COMMUNICATIONS SEQUENCE

The exchange of data between the processor and a UNISCOPE terminal is accomplished by a system of interrogation (polling) by the processor and response by the UNISCOPE terminal. If a message is not ready for transmission when the terminal is polled, the UNISCOPE terminal reacts to the poll with a no-traffic message. The control of traffic within the system is exclusively the role of the processor. The terminal functions only in response to a poll. A description of data flow for messages follows. The term "transmission" is used to describe the activity between the processor and the terminal via the terminal's communications interface.

### 2.5.1. Data Flow for Messages to Terminal

Messages sent to the terminal are received by the UNISCOPE terminal input/output (I/O) section in a bit-serial format. If operation is synchronous, each message is examined for SYN preamble characters to establish the character sampling intervals. Four consecutive SYN characters are recommended to achieve synchronization. In both synchronous and asynchronous operation, the received bits of the complete message are shifted in a bit-serial mode into data registers in the I/O section of the UNISCOPE terminal and they are checked for correct parity. If parity is correct, the text is entered in display storage and ultimately is displayed. See Figure 2-10 for format.

	HEADING					TEXT	TRAILER		
FOUR IDLE CHARACTERS*	R	S	D		COMMUNICATION CONTROL CHARACTERS	MESSAGE TO TERMINAL: CURSOR ADDRESS	TEXT	E T X	B C C
SYN	I	I	I	S					
SYN	D	D	D	T					
SYN	/	/	/	X		MESSAGE FROM TERMINAL: SOE ADDRESS OR HOME ADDRESS IF NO SOE ON SCREEN			
SYN	H	G	G						
	D	D	D						
*Applicable only for synchronous systems	Address Identifier			Supervisory Sequence					

2184

Figure 2-10. Message Format

### 2.5.2. Data Flow for Messages From Terminal

Messages sent from the terminal are composed on the CRT display by the operator, who positions the cursor and enters the desired data from the keyboard. Then the operator presses the TRANSMIT key and the terminal waits for a poll message. When the TRANSMIT key is pressed, the keyboard is locked (that is, disabled; it is not mechanically locked) and the traffic-ready condition in the UNISCOPE terminal is established. After a brief interval, the terminal will be polled by the processor. (Waiting time is a function of the programming and the activity of the processor; normally, there would be only a momentary delay.) Upon being traffic polled, the UNISCOPE terminal will assume the transmit mode and will generate a message with a format as shown in Figure 2-10.

### 2.5.3. Message Format

If a synchronous system is used, the message is preceded by four synchronous idle (SYN) characters (Figure 2-10).

The heading begins with the start-of-heading character (SOH) and is ended by STX or ETX. It contains a 3-level address identifier for routing and handling of the message, followed by any required communications control characters. The address is made up of RID (remote identifier), SID (station identifier), and DID (device identifier) characters for the three levels. A GID (general identifier) character can be used in one or more of the 3-level positions (3.3).

The supervisory sequence follows the address. It is made up of one or more communications control characters or sequences and is used for control functions such as acknowledgments, status indications, and device selection controls.

The text of the message contains the information to be transmitted. It begins with the start-of-text character (STX) and ends with the end-of-text character (ETX).

In a message to the terminal, the text begins on the terminal screen where the cursor is positioned. The cursor positioning sequence within the text of the message (either before text begins and after STX, or during the actual text) positions the cursor each time it is encountered within the text. If cursor control characters are not present, the text simply follows the cursor from left to right along the lines of the screen, beginning at its position when the text transmission starts, and continuing to the end of each line as long as data continues. In this situation, when the last space on the screen is filled, the cursor moves to the home position and starts across the first line. Any data on the first line is overwritten.

In a message from the terminal, the text sent from the terminal begins with the start-of-entry character (SOE) on the screen (or home position if no SOE is used) and continues until the cursor is reached. The address (screen position) of the SOE character (or home coordinates if no SOE is used) is given at the start of the message.

The processor can insert the SOE character in its text, such as at the end of a question to the terminal. Then, with the cursor allowed to remain in its normal position to the right of the SOE, the operator need only enter the answer and press the TRANSMIT key without being concerned about either the SOE or the cursor, and his answer will automatically follow the processor-placed SOE character.

The message is concluded with the end-of-text character (ETX), followed by the block check character (BCC). BCC is a longitudinal parity character used as an aid in error detection. For messages to the processor, the terminal generates and adds character parity to every 7-bit code transmitted. The terminal checks for character parity in the message from the processor. Character parity for synchronous transmission is odd (odd number of 1 bits); for asynchronous transmissions, it is even (even number of 1 bits). BCC begins with, but does not include, SOH and continues through and including ETX. The character parity bit of the BCC character itself is the same sense as the character parity of the text characters (odd for synchronous transmission, even for asynchronous transmission). (Refer to 3.5.6.)

A contiguous pair of end-of-transmission characters (EOT) is used to indicate a no-traffic-without-acknowledge response to a poll. (Normally built to include the ETX and BCC with the EOT characters, the UNISCOPE terminal has a strap option that deletes the ETX and BCC codes.) After transmission, the UNISCOPE terminal reverts to the receive mode, awaiting the next message from the processor.

In Figure 2-10, the characters SYN, SOH, STX, and ETX are actually transmitted characters, whereas the others (RID, SID, DID, GID, BCC) merely represent the types of code in the transmitted message.

#### 2.5.4. Message Acknowledgment

Message codes which acknowledge the receipt of a previous message are used in both processor and terminal messages to increase the reliability of the system by detection of any missing messages or parts of messages resulting from equipment malfunctions that might have occurred during transmission.

A message from the terminal is acknowledged by a control character code (DLE 1) in the supervisory sequence (Figure 2-10) of a poll from the processor. If the acknowledgment is missing and the terminal is expecting one, the terminal will respond with a reply-request message which asks the processor to either acknowledge the receipt of the original message or request retransmission of the original message by a retransmission-request message to the terminal. If locked, the terminal keyboard remains locked when the reply request and retransmission request are in effect, and it becomes unlocked when the text sequence is completed. A strapping option can be made that will unlock the keyboard on receipt of an acknowledgment.

A text message from the processor is also acknowledged by the terminal (via poll response) with an acknowledge code (DLE 1). The terminal acknowledgment is then acknowledged by the processor. The system can be programmed so that the processor will automatically send a message again when the text message is not acknowledged by the terminal.

### 2.5.5. Keyboard Functions and Octal Codes

Table 2-2 lists both uppercase and lowercase key labels (64-character or 96-character keyboards with shift) and the corresponding octal codes representing each key function label.

Table 2-2. Keyboard Functions and Corresponding Transmitted 7-Bit Octal Codes

Key Function	Octal Code	Key Function	Octal Code	Key Function	Octal Code
A	101	Q	121	#	043
a	141	q	161	\$	044
B	102	R	122	%	045
b	142	r	162	&	046
C	103	S	123	'	047
c	143	s	163	(	050
D	104	T	124	)	051
d	144	t	164	=	075
E	105	U	125	.	055
e	145	u	165	~	176
F	106	V	126	^	136
f	146	v	166		174
G	107	W	127	\	134
g	147	w	167	/	140
H	110	X	130	@	100
h	150	x	170	{	173
I	111	Y	131	}	133
i	151	y	171		137
J	112	Z	132	+	053
j	152	z	172	:	073
K	113	0	060	*	052
k	153	1	061	:	072
L	114	2	062	}	175
l	154	3	063	}	135
M	115	4	064	<	074
m	155	5	065	.	054
N	116	6	066	>	076
n	156	7	067	.	056
O	117	8	070	?	077
o	157	9	071	/	057
P	120	!	041	Space	040
p	160	"	042		

## 2.6. UNISCOPE TERMINAL AUXILIARY INTERFACE

The auxiliary interface provides a bit-parallel (7-bit ASCII coding), character-serial, request-acknowledge interface with which auxiliary devices can use the buffer storage and data entry-edit capability of the UNISCOPE terminal. The interface is capable of operating at speeds up to 30,000 characters per second (at distances up to 200 feet). The auxiliary interface permits up to eight auxiliary devices (up to 12 device addresses) to be connected to the terminal in a multidrop configuration. Data flow is half-duplex with a separate input and output data bus provided. Operation may be online under automatic selection and control by a processor or offline under operator control; or a combination of these two operations may be used.

The term "transfer" is used to describe the input/output activity between the terminal and an auxiliary device via the terminal's auxiliary interface. For auxiliary interface activity, input and output are relative to the terminal.

### 2.6.1. Functional Operation

A general discussion of the functional operation of the auxiliary interface (AI) follows. It includes information related to selection, data transfer, and status reporting. In the following paragraphs, refer to Table 2-3 for the ASCII equivalent of the octal codes used.

#### 2.6.1.1. Selection

Selection is the process by which a device becomes the current user of the AI bus. Selection may be either automatic (under processor control) or manual. In processor selection, the device address or identifier (DID) character is placed on the AI output data lines and the AI places a signal on the external function (EF) line. All devices connected to the interface go through a deselection sequence, and then the addressed device is selected and returns a status character using the input data lines and input data request (IDR) signal. As part of the processor selection sequence, AI input and output are terminated. Processor selection may be accomplished via either a poll or a text message containing a specific DID.

#### 2.6.1.2. Device Identifier (DID)

The DID is a 7-bit character selected from column 7 of the ASCII code chart (Figure 2-4). Of the 16 characters included in column 7, only 12 (octal 163 through octal 176) are available as device addresses. Octal 160 (p) has been designated as a general device identifier (GID). Octal 161 (q) has been designated as providing a means of getting a text message to the UNISCOPE terminal and/or terminating AI activity without deselecting the auxiliary interface device (AID). Octal 162 (r) both terminates AI activity and deselects the AID. Octal 177 (delete) is used with the keyboard PRINT key.

#### 2.6.1.3. DID Register

The DID register is the AI register that holds the DID used during a processor-controlled AI activity. During online operation, the specific DID character of the address of any processor-originated message addressed to the terminal containing the AI is placed in the DID register of the addressed terminal's AI. (The GID is never placed in this register.) Whenever the keyboard PRINT key is pressed, the delete character (octal 177) is loaded into the DID register. Also, the DID register provides the DID character of the address of all DID-containing messages returned to the processor from the UNISCOPE terminal.

#### 2.6.1.4. Auxiliary Interface Input Enable

When the AI is in an enabled state for input, the AI monitors and responds to the IDR line. If it is not enabled, no input operation through the AI is possible except for transfer of a single status code immediately following a selection sequence.

#### 2.6.1.5. Auxiliary Interface Output Enable

When the AI is in an enabled state for output, the interface monitors and responds to changes in the output data request (ODR) line. If it is not enabled, no output operation through the AI is possible except for the selection sequence and end-of-memory reporting.

Table 2-3. ASCII/Octal Code Conversions

Character	7-Bit ASCII Code (8-Level)	Octal Code	Character	7-Bit ASCII Code (8-Level)	Octal Code	Character	7-Bit ASCII Code (8-Level)	Octal Code
NUL	0000000	000	+	0101011	053	V	1010110	126
SOH	0000001	001	-	0101100	054	W	1010111	127
STX	0000010	002	.	0101101	055	X	1011000	130
ETX	0000011	003	,	0101110	056	Y	1011001	131
EOT	0000100	004	/	0101111	057	Z	1011010	132
ENQ	0000101	005	0	0110000	060	[	1011011	133
ACK	0000110	006	1	0110001	061	\	1011100	134
BEL	0000111	007	2	0110010	062	}	1011101	135
BS	0001000	010	3	0110011	063	^	1011110	136
HT	0001001	011	4	0110100	064	_	1011111	137
LF	0001010	012	5	0110101	065	`	1100000	140
VT	0001011	013	6	0110110	066	a	1100001	141
FF	0001100	014	7	0110111	067	b	1100010	142
CR	0001101	015	8	0111000	070	c	1100011	143
SO	0001110	016	9	0111001	071	d	1100100	144
SI	0001111	017	:	0111010	072	e	1100101	145
DLE	0010000	020	;	0111011	073	f	1100110	146
DC1	0010001	021	<	0111100	074	g	1100111	147
DC2	0010010	022	=	0111101	075	h	1101000	150
DC3	0010011	023	>	0111110	076	i	1101001	151
DC4	0010100	024	?	0111111	077	j	1101010	152
NAK	0010101	025	@	1000000	100	k	1101011	153
SYN	0010110	026	A	1000001	101	l	1101100	154
ETB	0010111	027	B	1000010	102	m	1101101	155
CAN	0011000	030	C	1000011	103	n	1101110	156
EM	0011001	031	D	1000100	104	o	1101111	157
SUB	0011010	032	E	1000101	105	p	1110000	160
ESC	0011011	033	F	1000110	106	q	1110001	161
FS	0011100	034	G	1000111	107	r	1110010	162
GS	0011101	035	H	1001000	110	s	1110011	163
RS	0011110	036	I	1000011	111	t	1110100	164
US	0011111	037	J	1001010	112	u	1110101	165
SP	0100000	040	K	1001011	113	v	1110110	166
!	0100001	041	L	1001100	114	w	1110111	167
"	0100010	042	M	1001101	115	x	1111000	170
#	0100011	043	N	1001110	116	y	1111001	171
\$	0100100	044	O	1001111	117	z	1111010	172
%	0100101	045	P	1010000	120	{	1111011	173
&	0100110	046	Q	1010001	121		1111100	174
'	0100111	047	R	1010010	122	}	1111101	175
(	0101000	050	S	1010011	123	~	1111110	176
)	0101001	051	T	1010100	124	DEL	1111111	177
*	0101010	052	U	1010101	125			

### 2.6.1.6. PRINT Key

PRINT is the only key on the UNISCOPE terminal keyboard that permits manual operation of the AI. To initiate AI output manually, assuming the device address has been selected, the operator places the SOE symbol at the beginning of the data to be transferred and the cursor at the end of the data. If no SOE symbol is used, the data is delineated by the home position and the cursor. In either case, the operator then presses the PRINT key, which locks the keyboard and initiates the output transfer via the AI. During the output transfer, the SOE symbol, its address, and all nonsignificant spaces are suppressed. To initiate AI input manually, assuming the device address has been selected, the operator positions the cursor over the SOE symbol and presses the PRINT key; the keyboard is locked and the input process via the AI is initiated. The input data appears on the screen beginning either at the operator-selected cursor position or at a cursor position selected by the input device.

### 2.6.1.7. Print Code

The print code (DC2) is normally used by the processor or an AID (for offline operations) to initiate input and output operations via the AI. The transfers are handled in a manner analogous to those initiated by the PRINT key.

### 2.6.1.8. End-of-Memory Output

The end-of-memory output (EOMO) character (ETX plus an EF) is issued when the last character in the output message (the character under the cursor) has been provided to the auxiliary interface during an output operation.

### 2.6.1.9. End-of-Memory Input

The end-of-memory input (EOMI) character (STX plus an EF) is issued when the last character position in the terminal storage is filled or when the AI has received an ETX character.

### 2.6.1.10. Print-Transparent Code

When the print-transparent code (ESC DC2) is used by the processor or by an AID (in offline operation) to initiate input data transfers, input operation is identical to that initiated by the print code. When the print-transparent code is used relative to AI output, the cursor return code normally introduced in the output data by the UNISCOPE terminal at the end of each display line is suppressed. Otherwise, the output operation is analogous to that initiated by the print code. It should be noted that the print-transparent code cannot be initiated directly from the UNISCOPE terminal keyboard.

## 2.6.2. Operation

There are three types of AI operation: online, offline, and combined. Online operation refers to processor-initiated selections and processor-initiated print (or print transparent) commands. Offline operation refers to manually initiated selections and manually initiated print commands. Combined operation refers to processor-initiated selection and manually initiated print commands.

### 2.6.2.1. Online Operation

Certain items of information regarding AI and AID status are provided to the processor (via the UNISCOPE terminal poll response) as a result of processor-initiated selection sequences and/or processor-initiated print and print-transparent commands.

The AID status information is conveyed by the presence (and by the absence) of a status response at a unique reporting time. The status response is composed of a status code on the AI input data bus in conjunction with an IDR.

AID status codes and their meanings are listed in Table 2-4.

Table 2-4. Status Codes

AID to AI Input Bus				Description	AI to Processor
Bit 4	Bit 3	Bit 2	Bit 1		
X	X	1	1	Device ready	DLE >
X	X	1	0	Error 1 (meaning defined by each device)	DLE <
X	X	0	1	Error 2 (meaning defined by each device)	DLE :
X	X	X	X	No status response	DLE =

For any processor-initiated selection, the AI will interpret the AID condition according to the appropriate status code. The first three codes in Table 2-4 are actually responses from the AID, consisting of the status codes shown plus an IDR. The fourth is produced by the AI if the absence of response (no IDR) exists at the status reporting time (the selecting DID character time). Although not considered by the AI, the no-response status code from the AID would normally be all 0's.

If the selection is accompanied by text but with neither a print nor print-transparent command, the AID status will be returned to the processor on the first valid poll following the selection. In this case, or if the selection has been done via a poll message, the poll response will convey the AID status by including the appropriate 2-character response (bits 1 and 2 in Table 2-4).

Assuming that device-ready status (DLE >) has been provided, the subsequent receipt of a processor-initiated print or print-transparent command in a correct message will cause a busy status condition (WABT) during the AI data transfer operation. (Slow polling rates and/or fast data transfers can create situations where THRU could be returned without WABT first being returned to the processor.)

In terms of responses to the processor, all of the status conditions are mutually exclusive; that is, only one is available from a single terminal at any time. The four AID status conditions are reported to the processor (via a poll response) with the 2-character sequences shown in Table 2-4. Busy and through conditions are reported as WABT (DLE ?) and THRU (DLE ;). With the exception of WABT, these online status conditions are reported once and only once (unless retransmit is employed by the processor) in response to the first valid poll message following the activity that created the condition. In accordance with the UNISCOPE terminal communications control procedures, the WABT is returned only once in response to a general poll but will be returned in response to every specific poll (specific RID and SID, general DID) for the duration of the busy condition.

When sent to an AI-equipped terminal, any processor-originated message with a DID other than octal 160 causes a change in online status. A DID of octal 161 clears the AI status without deselecting the AID. A DID of octal 162 or greater requires (at least) that an AI selection sequence follow. Thus, polls following selection should use the general DID (octal 160) to avoid such conditions as losing or changing status or inadvertent termination of AI data transfers. It is possible to get another 2-character AI response, DLE 8. The circumstances precipitating this response are discussed in 2.6.2.4.

The four 2-character responses shown in Table 2-4 are based on the combinations shown for bits 1 and 2 of the AID-to-AI status word. Bits 3 and 4 convey other information.

Bit 3 of the status word from the AID identifies an input AID (1) and an output AID (0).

Bit 4 provides the AID with the alternative of attempting or not attempting a print or print-transparent operation if error 1 or 2 status is returned. If bit 4 is a 1, a print or print-transparent operation will be attempted despite the error status condition. In this case, the busy state is entered and the AID must then decide to honor the data or stop requesting it. The AI successfully completing the data transfer results in terminating the busy condition and entering the THRU condition, as with a normal operation (print or print-transparent with device-ready status). Otherwise, the WABT is sustained, with cessation of data flow prior to the completion of transfer, until cleared either via processor or manual control. If bit 4 is a 0, on error 1 or 2 status, the print or print-transparent is dumped. In this case, the data transfer is not allowed and no AI WABT/THRU indications are made available. The error status is, however, available for reporting provided no activity occurs to change the AI online status. Bit 4 is ignored by the AI on device-ready status. If a no-response status has resulted, it too will precipitate a print or print-transparent dumping and will be made available for reporting.

It should be noted that WABT and THRU refer to the AI condition and do not necessarily correspond to the AID condition. As an example, the last output character of a message could send an AID into an extended sequence (such as search or rewind) and THRU could be generated, indicating completion of the AI data transfer, before the AID had completed its operation. Thus, the significance of WABT/THRU must be interpreted in light of the type of AID involved and the type of operation being performed.

Both output and input online data transfers are initiated by a print or print-transparent command.

If an AI output data transfer is occurring, AI activity will be terminated when EOMO is supplied. After the last output character (the character under the cursor) is supplied to the AID, the EOMO code (ETX) is placed on the AI output data bus and an EF signal is generated.

The AI output data stream may include any or all of the following:

- Columns 2 through 7 of the ASCII code chart.
- All control codes from columns 0 and 1 of the ASCII code chart to which the UNISCOPE terminal is transparent; that is, control codes that are simply accepted by the UNISCOPE terminal and placed in its storage (the SOE character and its related output coordinates are stripped off by the AI, however).
- The cursor return code (CR of the ASCII code chart).

If an input AID is involved, an output data transfer is simulated by issuing a print or print-transparent command with the SOE as the only output character. This simulated output results in the generating of the EOMO code and the EF signal. The AI is then enabled for input; no ODR is required during the simulated output since the character under the cursor is an SOE. Input data can then be supplied until an ETX is included in the input data stream, at which time the AI will exit input mode and generate an EOMI. The EOMI code, STX, is placed on the AI output data bus and an EF signal is generated.

An EOMI is also generated if the last (end of screen) character position of the UNISCOPE terminal storage is filled. The terminal cursor will wrap (go to the beginning of the screen), and the terminal will continue to accept data until an ETX is supplied in the input data stream.

On input, the AI passes all codes from columns 0 through 7 of the ASCII code chart except NUL and ETX. All other codes are transferred to the UNISCOPE terminal. The terminal, in turn, determines whether the code should be acted upon or stored. Thus, all normal UNISCOPE terminal control code sequences, including cursor addressing, line insert/delete, and character insert/delete, are available for use by input AIDs via the AI. These capabilities allow an input AID to position the cursor and perform any of the previously discussed control functions that the processor might perform.

Similarly, the shift-out (SO) and shift-in (SI) codes are available for use in the manner appropriate to the UNISCOPE terminal; that is, in a protected format terminal, the SO and SI would designate the beginning and ending, respectively, of a protected field.

The format of control, data, and sequences input via the AI is identical to that specified for processor data transmitted to the UNISCOPE terminal between the STX and ETX codes of the processor-initiated text message.

Data transfers to and from the UNISCOPE terminal storage may involve either the communications interface (2.4) or the AI, but not both simultaneously. The communications interface is susceptible to the receipt of any of three types of messages while a print or print-transparent function is occurring. The three types of messages are processor message waiting, polls, and output text; they are handled as follows, assuming a general DID (octal 160) in each case.

1. The processor message waiting (BEL) output causes the MESSAGE WAITING indicator to light and the audible alarm to sound.
2. The poll solicits AI status (WABT/THRU) or UNISCOPE terminal status, as appropriate.
3. The output text message is immediately terminated and, hence, not put on the screen.

If the DID of the three possible messages was other than octal 160, the print operation would be terminated and the current AI online status cleared. Furthermore, if the DID were octal 163 or greater, a selection sequence based on the new DID would occur, with appropriate online status resulting. If octal 162, a selection sequence based on 162 (not a legal AI DID) would occur with a no-response online status resulting; if octal 161, the currently selected AID will remain selected (no selection sequence will occur). If output text accompanied a DID other than 160, the text would not be accepted unless a number of NUL characters were supplied by the processor between the DID and STX characters. The nulls allow time for the print sequence to be terminated. The text would then go on the screen with online status appropriate to the message resulting. A worst-case delay of 20 milliseconds should be provided, although precise knowledge of the terminal's condition permits less time in suitable circumstances.

The UNISCOPE terminal requires the keyboard to be locked when the PRINT key is pressed or when the print-transparent code is detected. It is the responsibility of the AI to clear such lock conditions at the proper time, although the keyboard may remain locked because of other UNISCOPE terminal operational requirements. For output, the AI unlocks the keyboard when the EOMO occurs. On input, the keyboard locks during the simulated output and unlocks at its completion, immediately relocks and then unlocks when the inputting AID provides an ETX. Also, any new selection clears any AI-induced keyboard lock, as does the UNISCOPE terminal WAIT switch.

Any transmit (DC1, ESC DC1) or print (DC2, ESC DC2) command sequence that is included in the AI input data stream will terminate transfers between the AI and the UNISCOPE terminal. Since the detection of these codes is done by the UNISCOPE terminal, not by the AI, an ETX is still required to terminate the AI input mode and to unlock the keyboard. (The print sequences, of course, can reinitiate AI activity.) Thus, it is required that print or transmit commands, if used in the input data stream, immediately precede ETX, the normal ending character for all input data streams.

### 2.6.2.2. Offline Operation

Offline operation of the AI is functionally identical to online operation with the following exceptions:

- Selection is accomplished manually, and no selection status reporting either to the AI or to the processor is involved.
- Neither WABT nor THRU (DLE ? and DLE ; respectively) is generated as a result of the print operation.

- Polls with a general DID (GID) result in no-traffic responses despite AI offline activity. Other UNISCOPE terminal conditions may override the no-traffic response, however. The DID of the no-traffic response is octal 177 if the PRINT key has been pressed.
- Input is enabled following EOMO regardless of whether a real or simulated output occurred. If an IDR signal is present, the keyboard will be locked and input enabled until an ETX is provided in the input data stream.
- Data transfers are initiated via the PRINT key.

### 2.6.2.3. Combined Operation

Combined operation involves essentially offline data transfer, although the following unique characteristics exist:

- Combined operation involves processor selection with the resulting selection status (DLE >, DLE <, DLE :, DLE =) being made available to the processor via a poll response. Poll responses are dependent upon the time relationship between the arrival of the poll, the actual selection (poll versus specific DID and text), and pressing of the PRINT key. If selection status has not yet been picked up by the processor and the PRINT key has not yet been pressed, a general DID (octal 160) poll will bring in the selection status. If the PRINT key had been pressed before the selection status was returned, the general DID poll response would be DLE 8 (2.6.2.4). A DID of 160 is necessary since any other DID would clear out the pending status and terminate the AI data transfer, if in progress. Furthermore, a DID of 162 or greater would initiate a new selection attempt, thus giving rise to new status. A DID of 161 in a poll would bring a response of DLE 8 or no-traffic, as described in 2.6.2.4. Polls with a DID of 160 following status indication responses will result in no-traffic responses, unless other UNISCOPE terminal conditions override, regardless of whether the AI is active.
- Pressing the PRINT key causes a 177 to be loaded into the DID register. Poll responses following the pressing of the PRINT key contain a DID of 177 until the processor changes the DID register (via a DID other than 160) or until the register is master cleared.
- Since AI data transfers are initiated manually via the PRINT key, no WABT/THRU generation occurs, as with offline operation.

### 2.6.2.4. DLE 8

Under certain conditions, the AI can return a DLE 8. Essentially, DLE 8 is generated in those circumstances where the communications interface expects to send status (selection status or THRU) and the AI has had the status cleared. Status is cleared when the PRINT key is pressed and/or when a DID of 161 is received. Nonselection polls following either condition result in a DLE 8 if status (either selection status or THRU) is owed. If status is not owed, a no-traffic response should result, unless overridden by other UNISCOPE terminal conditions.

### 2.6.3. Device Requirements

The general requirements imposed on each device by the AI are:

- Each device must comply with the electrical and logical requirements of the auxiliary interface bus.
- Each device must be capable of connection in a daisy-chain sequence, preferably by a means that permits device removal without breaking the chain.
- Optionally, each device may provide manual select/deselect capability. When provided, this capability must be of the momentary variety; that is, manual selection can be overridden at any time by an automatic select and vice versa.

- Under the conditions previously specified, the AI provides EOMI and EOMO codes in conjunction with the EF signal. Use of these codes is optional for each device, but no device may interpret either of these as selection sequences since neither code is within the valid range of device addresses.
- As a matter of normal convention, the last character of any input data sequence should be ETX.

#### 2.6.4. UNISCOPE Terminal Operation Without AI

Since no DID register exists without the AI, the DID character of the address in any processor-originated message is ignored by the UNISCOPE terminal. The DID character of the address in any terminal-originated message will be 160.

### 2.7. DIFFERENCES IN UNITS

The major functional difference between the UNISCOPE 100 terminal and the UNISCOPE 200 terminal is the capacity of storage and display. Whenever one terminal is substituted for the other or both are mixed in a network, provision must be made for this difference in storage and display capacity. (When making such a substitution, refer to 3.3.5 for timing information and to 3.5.1 for cursor addressing information required for adjusting to the different terminal.)

The only other functional differences between the UNISCOPE 100 and UNISCOPE 200 terminals are those applying to operation with an auxiliary device. The larger display capacity of the UNISCOPE 200 terminal requires use of a different version of the tape cassette system. Differences between versions of the tape cassette system are covered in 2.8. The UNISCOPE 200 terminal also uses a different auxiliary interface (auxiliary interface feature 1247-01) that eliminates the cursor wraparound condition, as described in 4.2.2, and enhances the capability of the protected format feature in the tape cassette system, as described in 2.8.12. However, later-model UNISCOPE 100 terminals may be equipped with auxiliary interface feature 1247-01 and are then functionally identical to the UNISCOPE 200 terminal except for the display capacity. For specific information concerning incorporation of feature 1247-01 in UNISCOPE 100 terminals, contact your local Sperry Univac customer engineer.

#### 2.7.1. Differences in UNISCOPE 100 Terminal Series

Changes affecting the functioning of the UNISCOPE 100 terminal have been introduced at times and are noted by serial numbers of the units. When a display terminal system contains a mixture of old and new units, these functional differences must be taken into account.

**NOTE:**

*The functional characteristics related to a serial number series are based on the terminal configuration as it was shipped from the factory. Modifications made in the field can change these characteristics. For further information on specific units, contact your local Sperry Univac customer engineer.*

A serial split was made at serial number 6000, introducing the protected format capability. Units below this number do not have protected format capability. It is not necessary to use this capability (on units above serial number 6000), but if it is used, the differences in text format, cursor placement, and editing functions should be considered.

A second serial split was made at serial number 14,000 to incorporate a number of functional improvements. Three conditions that should be considered when using units below serial number 14,000 are:

1. When a UNISCOPE 100 terminal is turned on, a traffic indication (DLE 0) or a status indication can result.

2. If the terminal is in synchronism, control codes can be recognized and can cause internal UNISCOPE 100 terminal actions even though the terminal is not addressed. Thus, an ENQ (on the line) could cause the terminal to interpret a subsequent traffic poll as a status poll.
3. If a UNISCOPE 100 terminal receives a DNAK while in an EOT state, no response is sent and a processor timeout occurs.

Terminals between serial numbers 6000 and 14,000 place the keyboard lock code (DC4) into the terminal storage. This action can cause undesirable side effects and should be avoided by using an ESC DC4 code for the keyboard lock function.

When power is turned on, terminals with the protected format capability enabled may come up with random storage locations protected, including home position. This condition may make it impossible for the operator to place the cursor at the desired location. To clear this condition from the terminal, two approaches may be used:

1. A special BEL or function-key message can be provided which, when entered by the operator, will be interpreted by the software to do a cursor-to-home, erase-display sequence.
2. The operator can remove the location protection by pressing and holding the WAIT switch/indicator and the cursor scan-right key. This action cycles the cursor through storage, entering an unprotected character into storage at each protected location. With the protection thus removed, normal operation can then proceed.

### 2.7.2 Differences in UNISCOPE 200 Terminal Series

There are no serial-split differences in the UNISCOPE 200 terminal.

When the UNISCOPE 200 terminal is turned on, the cursor always appears in the home position and the screen is filled with unprotected characters; the terminal will not come up with random protected characters in any position. This condition occurs with or without the protected format capability in the terminal.

## 2.8. TAPE CASSETTE SYSTEM CHARACTERISTICS AND OPERATION

The tape cassette system is a high-volume storage device which records data, entered by way of a communications terminal, on either of two magnetic tape cassettes. The tape cassette system operations discussed in this manual apply only to data entered in or retrieved from the device by way of a UNISCOPE terminal.

Two versions of the tape cassette system are available for use with UNISCOPE terminals: tape cassette system types 0866-00 and -01, and types 0866-02 and -03. Types 0866-02 and -03 contain additional features designed for the increased display capacity of the UNISCOPE 200 terminal and the enhancements of the auxiliary interface (feature 1247-01) used with the UNISCOPE 200 terminal. UNISCOPE 200 terminals can be used only with tape cassette system types 0866-02 and -03. However, UNISCOPE 100 terminals capable of operating with types 0866-00 and -01 will also operate with types 0866-02 and -03.

### NOTE:

*In the following discussion, when the terms "cassette system" or "tape cassette system" are used without a type number, the information pertains to all types of cassette systems. The specific type number is used when applicable.*

## 2.8.1. Equipment Description

The cassette system consists of two tape transports, one set of controls and indicators, and associated electronic circuitry. Electronically, the transports share control circuitry and the circuitry which interfaces the total cassette system with the UNISCOPE terminal. However, each transport has its own 2-track read/write head and its own circuitry for beginning-of-tape and end-of-tape detection, address reporting, and status retention. Only one tape cassette can be selected and operated at a time, except during the optional edit and copy function when both cassettes operate in an alternate sequence as described in 2.8.14.1.

The tape cassettes (Figure 2-11) are self-contained, each loaded with 300 feet of 150-mil-wide, 1-mil-thick, computer-grade magnetic tape. They are self-threading units with a length of clear leader at each end of the tape. A small hole is punched in each end of the magnetic portion of the tape, approximately 17 inches from the clear leader. These holes mark the beginning and end of usable tape. The cassette has a write-protect capability to prevent inadvertent overwriting of recorded data. (See Figure 2-11.)

### 2.8.1.1. Standard Cassette System Features

The standard tape cassette system has the following basic capabilities:

- Write and read (store and retrieve) operation
- Character and block parity generation and checking
- Error detection, reporting, and recovery
- Tape positioning search

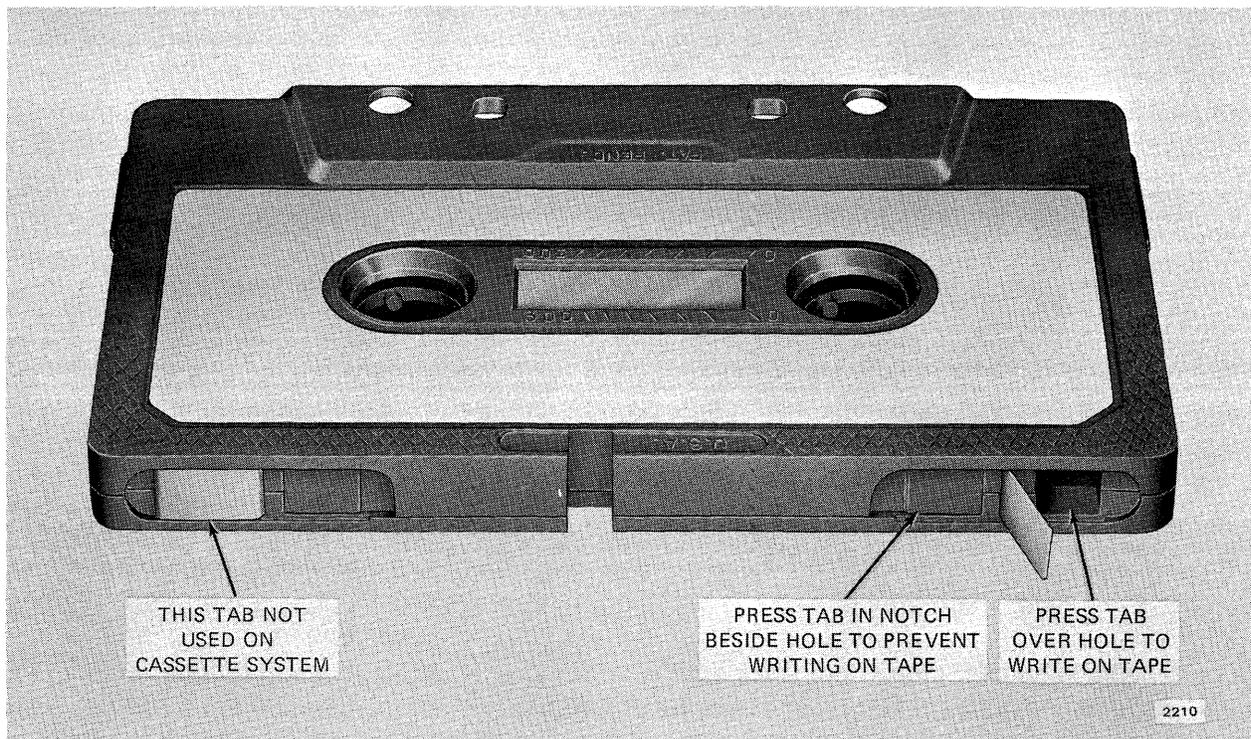


Figure 2-11. Tape Cassette

- Backward-one-block tape repositioning
- Address search
- Address reporting
- High-speed rewind

These capabilities are described in more detail in 2.8.4 through 2.8.9.

### 2.8.1.2. Optional Features

Several optional features may be added to enhance the basic tape cassette system capabilities. These features may be added in two groups: feature groups A and B for tape cassette system types 0866-00 and -01, and two corresponding groups, D and E, for tape cassette system types 0866-02 and -03.

- Feature group A includes:
  - The read-after-write feature, which allows character parity and block parity to be checked with a read performed as the block is being written
  - The protected format feature, which allows data to be written on tape as protected data
  - The list feature, an offline function which allows a printer to make a copy of the tape contents
  - The edit feature, an offline function which allows the operator to sequentially edit data on one tape and create a new tape containing the edited information, or to directly copy one tape to another.
- Feature group B includes all the features in group A, plus:
  - The record separator feature, which allows record separator characters to be written on a tape
  - The identifier search feature, which expands the standard address search capability to include a comparison of search identifier characters with actual data block characters
- Feature group D includes all features in group A, plus:
  - The HT (tab stop) control character feature, which allows the HT control character to be used simultaneously as the tab stop function and as a translation character for the protected format SO (shift out) or SI (shift in) function.
  - The print transparent feature, which enhances the list feature by allowing the printer to print tape data in line lengths determined by the printer format. (The cursor return function generated by the UNISCOPE terminal at the end of each line of data is suppressed, or becomes transparent, to the printer.)
- Feature group E includes all the features in groups A, B, and D, plus the copy-to-address feature, which provides the capability to terminate a copy mode at a specified preamble address.

The optional features are described in greater detail and programming considerations are discussed in Section 4.

## 2.8.2. Cassette System Configuration

The cassette system is connected to the UNISCOPE terminal through the auxiliary interface (2.6) and is designated by four DIDs. Each tape cassette transport uses two DIDs — one for each read channel and one for each write channel. Since there are 12 usable DIDs available with the auxiliary interface, up to 3 cassette systems may be attached to the auxiliary interface. However, if other auxiliary devices, such as printers, are connected to the auxiliary interface, fewer cassette systems can be connected. Only one auxiliary device may be selected at a time by the processor. If the equipment is intended for operation only under offline conditions, up to eight cassette units may be connected on the auxiliary interface.

If the optional list feature described in 2.8.14.1 is to be utilized with the cassette system, a printer should also be connected to the auxiliary interface.

The last unit in a chain of devices connected to the auxiliary interface must be equipped with a terminating resistor block, Sperry Univac part number 2807737-00.

## 2.8.3. Tape Cassette Interchangeability

The tape cassettes are generally interchangeable among cassette systems; a tape written in one cassette system may be read in another system. The only restriction is that a tape must be read on a UNISCOPE terminal of the same display screen format (or larger format) as the terminal on which the tape was written. For example, a UNISCOPE 100 terminal may have a format of 12 lines by 80 characters or 16 lines by 64 characters; if a 12 by 80 terminal is used to write a tape, that tape can be read only on a 12 by 80 UNISCOPE 100 terminal or a 24 by 80 UNISCOPE 200 terminal.

Table 2-5 lists all tape cassette interchangeability conditions.

### NOTE:

*If a tape is written in a smaller format size than that of the terminal used, the tape may be read on a terminal conforming to the written format. For example, a tape written from a 12 by 80 terminal, but written in a 64-character line length, may be read on a 16 by 64 terminal.*

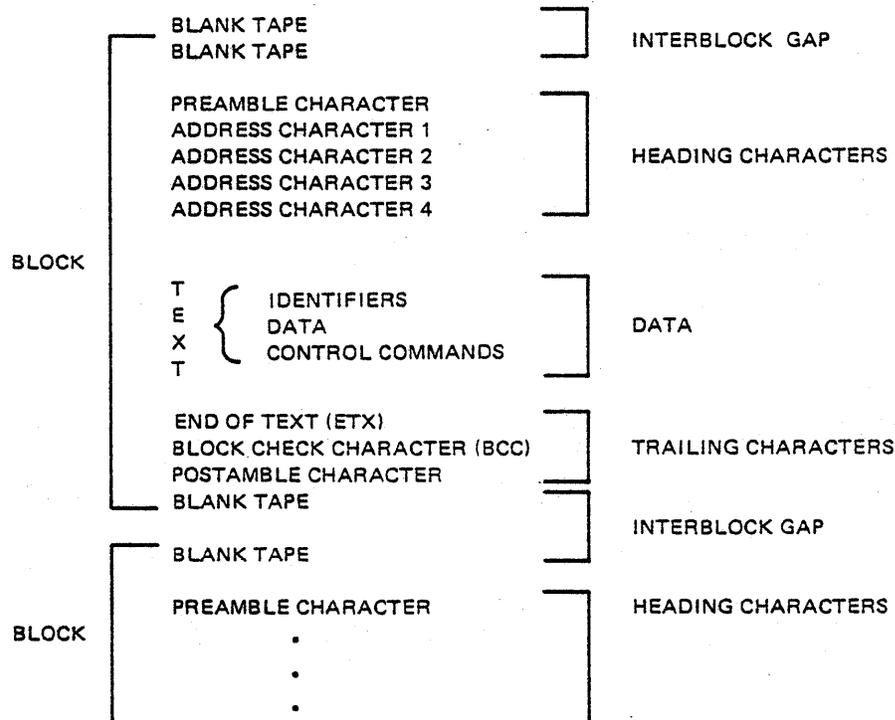
Table 2-5. Tape Cassette Interchangeability Conditions

Tape Written On	Can Be Read On	
UNISCOPE 100 terminal	UNISCOPE 100 terminal	UNISCOPE 200 terminal
16 by 64 format	16 by 64 format	24 by 64 format 24 by 80 format
12 by 80 format	12 by 80 format	24 by 80 format
UNISCOPE 200 terminal	UNISCOPE 100 terminal	UNISCOPE 200 terminal
24 by 64 format	None	24 by 64 format 24 by 80 format
24 by 80 format	None	24 by 80 format

## 2.8.4. Write/Read Implementation

### 2.8.4.1. Tape Format

The tape cassette system writes data in a specialized tape format, shown in Figure 2-12, to ensure reliable write and read operations.



2211

Figure 2-12. Tape Format

The heading characters, which are generated by the cassette system itself, consist of a preamble character used for internal synchronization and four address characters which indicate the tape-position address at the beginning of the block.

The text in any one block consists of all the data characters entered between the start-of-entry (SOE  $\triangleright$ ) symbol and the cursor ( $\square$  or  $\blacksquare$ ), or between the home position and the cursor if there is no SOE symbol, on the UNISCOPE terminal screen. The first 1 to 16 data characters may be used as identifiers in the optional identifier-search modes described in 2.8.7.

The trailing characters, also generated by the cassette system, consist of the end-of-text (ETX) character, a block parity check character, and a postamble character used for internal synchronization.

Only the data characters are seen on the UNISCOPE terminal screen (offline) and by the processor (online). The cassette system removes all the heading and trailing characters from the block during a read operation (except in the case of a report-address processor command, as explained in 2.8.9).

The pinch roller disengages at the end of each block and then reengages; the amount of tape passed in the pinch roller disengage/engage operation is the interblock gap. One-half of the interblock-gap tape is passed before the disengage operation and one-half after the engage operation.

#### 2.8.4.2. Selection

Selection is the process of designating one of the two tape transports for operation on the auxiliary interface bus and indicating whether a read or write operation is desired. In offline operation, the operator makes these selections by pressing the appropriate controls. In online operation, the processor sends a selection message. (Processor selection is covered in Section 4.)

#### NOTE:

*In the following paragraphs, only the data transfer activity between the cassette system and the auxiliary interface in the UNISCOPE terminal is discussed; processor and terminal activity is not. A detailed functional description of the auxiliary interface operation is contained in paragraph 2.6. The cassette system interface conforms with all the requirements of the auxiliary interface.*

#### 2.8.4.3. Write Operation

After the text has been entered in the UNISCOPE terminal and the desired tape transport and the write function have been selected, the output-data-request (ODR) line to the auxiliary interface is activated by the cassette system. A print command is sent to the auxiliary interface (the print command may be processor or manually issued), and the auxiliary interface performs the print command by presenting the first data character of the text message, plus an output-acknowledge (OA) signal, to the cassette system. When the cassette system receives the OA signal, the transport starts, the heading characters are written, and the data character is recorded.

Each subsequent data character in the text message is requested in the same way — by activating the ODR line, receiving one data character plus an OA signal from the auxiliary interface, and recording the character — until the auxiliary interface presents the end-of-memory-output (EOMO) code. This code indicates that the last data character in the text message (the character under the cursor on the terminal screen) has been supplied. The cassette system translates the EOMO code as an end-of-text (ETX) character, writes the ETX and the other trailing characters, and stops the transport.

If there is no intervening selection (that is, if a different tape transport or a different type of operation is not selected), the cassette system activates the ODR line again when the next print command is sent to the auxiliary interface. The next data character from the auxiliary interface is interpreted as a new write operation, and the sequence described in the preceding paragraphs is repeated to write another block.

If a parity or timing error occurs on a data transfer while the block is being written, the cassette system goes through an error exit and recovery sequence (described in 2.8.6) before another operation can be performed.

#### 2.8.4.4. Read Operation

After the desired tape transport and the read function have been selected, the output-data-request (ODR) line to the auxiliary interface is activated by the cassette system. A print command is sent to the auxiliary interface and, as a result

of the command, the auxiliary interface presents the end-of-memory-output (EOMO) code to the cassette system. The cassette system then deactivates the ODR line, presents one or more nondata characters (2.8.4.5) to the auxiliary interface, and activates the input-data-request (IDR) line. The auxiliary interface returns an input-acknowledge (IA) signal in response to the nondata character. Upon receipt of the IA signal, the cassette system deactivates the IDR line, starts the transport, reads the heading characters, obtains the first data character of the block from tape, and presents the character plus another IDR signal to the auxiliary interface. The auxiliary interface returns an IA signal to indicate that the character was received.

Each subsequent data character in the block is obtained from tape in the same way — by reading a character and presenting it to the auxiliary interface, activating the IDR line, and receiving an IA signal — until the entire data block is transferred. Then the cassette system sends the ETX code to the auxiliary interface. After ETX transmission, the IDR line is not activated again; the cassette system reads the trailing characters and then stops the transport in the interblock gap.

If there is no intervening selection, the next print command is interpreted as a new read operation and the sequence is repeated to read another block.

Parity errors detected in a read operation are processed as described in 2.8.6.

In addition to the sequential data-block read operation just described, the cassette system can also perform three different selective-read operations. Two of these operations (search and backward-one-block tape repositioning) can be commanded either by manual or processor control, and the other (report tape address) can be commanded under processor control only. The functional operations of the cassette system as a result of these unique commands are described in 2.8.7, 2.8.8, and 2.8.9, respectively.

#### 2.8.4.5. Cursor Positioning

Cursor positioning occurs on read and search operations (except on @-type search) to clear the terminal screen of any data displayed from the previous data block (or only the unprotected data if the terminal and cassette system are so equipped). When a print command is sent to the cassette system to initiate a read or search operation, any data on the screen is presented to the cassette system, a character at a time. However, the cassette system recognizes only the end-of-memory-output (EOMO) code which occurs at the end of the screen data, or character string. This sequence (EOMO code after characters) signals the cassette system that the cursor is to be repositioned; in response, the cassette system presents nondata characters specifying a cursor-to-home and erase-screen sequence before reading the first data character from tape. (The erase-screen character in the cassette system is strappable; it can be selected either to erase only the unprotected data display (ESC a) or to erase both protected and unprotected data (ESC m).)

If an EOMO code alone is received by the cassette system, the code implies that the cursor has been placed over the SOE and therefore that cursor positioning has not been specified; in this case, the nondata character supplied before read is the NUL code. If the cursor is placed over the SOE and a read initiated, the screen is not erased but data is entered beginning at the cursor/SOE position.

#### 2.8.5. Parity Generation and Checking

The cassette system utilizes both character parity and block parity to check the accuracy of data transfers from the auxiliary interface. Character parity is written after each character, and block parity is accumulated and then written or compared at the end of the block.

In a write operation, the 7-bit data characters presented by the auxiliary interface are modified by the cassette system to include an eighth bit so that the sum of the recorded character has odd parity. The cassette system also generates a block parity check character (BCC) by logically summing (without carry) each independent column for each character in the data block, from the first data character through and including the ETX character, and recording the complement of the total (also with odd character parity). The BCC is recorded immediately after the ETX character.

In a read operation, the cassette system verifies character parity as each character is read from tape. Block parity is accumulated and compared with the recorded BCC.

Parity error handling is covered in 2.8.6.

### 2.8.6. Error Detection, Reporting, and Recovery Operations

Parity and timing errors (collectively referred to as data errors) and improper-selection errors are detected by the cassette system and identified with a special error code in the data and with audio and visual indications in the cassette system. The system error exit and recovery procedures prevent parity and timing errors from passing through the auxiliary interface in a read operation. However, error status is reported to the auxiliary interface on a selection. In online operation, the UNISCOPE terminal translates the error code from the auxiliary interface into a code form recognizable by the processor. When data is sent to the processor, this code is included with the data so that software programs can decide on the proper recovery action to take. (Refer to 4.8 for error recovery procedures.)

The special error code, which is available to accommodate various user applications, is strapped in at the user site by the Sperry Univac customer engineer.

#### 2.8.6.1. Parity Errors

If incorrect parity is encountered in a read operation, the cassette system immediately supplies the special error code to the auxiliary interface in place of the erroneous character and retains the error status. The read is discontinued and the tape is positioned at the next interblock gap; then the transport stops, the CHECK indicator on the cassette system lights, and the alarm sounds. With the next selection of that transport, the error status is reported to the auxiliary interface and then cleared, the indicator goes out, and the alarm stops. Upon the next selection, the designated operation is performed in the normal manner.

#### 2.8.6.2. Timing Errors

If a timing error occurs during a write operation (that is, if a character does not appear within the area of tape intended for that character), the special error code is written on the tape and is immediately followed by the normal block termination sequence. If a timing error occurs during a read operation, data transfer stops and the tape is positioned at the following interblock gap.

When the interblock gap following the error block is reached, the transport stops and the error status is retained. The error reporting and recovery sequence is the same as for parity errors.

#### 2.8.6.3. Improper-Selection Errors

If a new selection of the cassette system is attempted by the processor while a block is being written or read, the operation is interrupted and the cassette system reports a data error. When the interruption occurs during a write operation, the character being written is completed and then the special error code is written, followed by the normal block termination sequence. When the interruption occurs during a read operation, data transfer stops and the tape is positioned at the following interblock gap.

At the interblock gap following the block in which the interruption occurred, the transport stops and the audio and visual error indications occur. The error recovery sequence depends upon which transport is selected next; refer to 4.8 for a detailed description.

#### 2.8.6.4. Tape Problems

If there is a bad spot (defect) on the tape, it will be impossible to record or retrieve data on that portion of the tape, and attempted use will result in a data error with indications as described in 2.8.6.1. An error will also be reported if the tape is inserted wrong or if there is contamination on the tape. (Any dirt or lint should be carefully removed from the tape with a cotton swab.)

#### 2.8.6.5. Read-After-Write Errors

Units equipped with and using the read-after-write feature (2.8.13.1) will detect errors in data immediately after each character is written. When an error is detected, the CHECK indicator immediately lights and the audible alarm immediately sounds, but writing of data and immediate reading of each character continues to the end of the block; then the tape is positioned in the interblock gap at the end of the block, the transport stops, and the error status is retained. The error reporting and recovery sequence is the same as for parity errors (2.8.6.1 and 4.8.3).

#### 2.8.7. Search

Four types of search can be commanded, in either online or offline operation, to position the tape at a specified location. Combined with the backward-one-block and online report-address functions (2.8.8 and 2.8.9, respectively), the search modes give the cassette system an extremely versatile data manipulation and retrieval capability. The search function also provides an online, high-speed rewind capability. A processor command to search to address zero initiates the rewind.

Online, the search modes are initiated by a special processor control command followed by a search information statement. Offline, the search function is "commanded" by pressing a control on the cassette unit and then entering the search information statement on the UNISCOPE terminal screen.

Except in mode @, the block of data matching the search statement is read as soon as it is located and is displayed on the terminal screen. Only the one block will be read with each search command; subsequent blocks must be read by issuing a read command.

The mode @ and mode A searches described in 2.8.7.1 and 2.8.7.2 are standard cassette system functions. Modes B and C (2.8.7.3 and 2.8.7.4) are optional features offered as part of a feature group package.

##### 2.8.7.1. Mode @ Search

In this search mode, the tape is driven backward or forward at high speed to the tape position indicated by the address supplied in the search information statement. The transport then stops; no automatic data read takes place until a read selection is made and a read operation is commanded. (Also, no write will occur unless a write operation is commanded.)

##### NOTE:

*In a mode @ search, if a write operation is selected after the tape stops, any data on tape at the search location will be erased when the write function is selected.*

Based solely on the physical tape address count kept by the cassette system and displayed on the face of the unit, this search positions the tape within the area of the supplied address but is not as accurate in positioning as the other search modes. The address count is determined by an optical tachometer inside the cassette system, with one tachometer count being defined as one-eighth of a revolution of the tachometer wheel on the supply reel motor. Therefore, the physical length of tape associated with a tachometer count will vary from the beginning of tape to the end, and the tape position indicated by the address may be anywhere within the space represented by one division of the tachometer.

### 2.8.7.2. Mode A Search

In this mode, the tape is driven forward or backward at high speed to an address count of 20 preceding the address indicated in the search information statement. The search then continues at read speed to compare the supplied address with the address in the block heading. When the addresses match, a cursor-to-home and erase-screen operation is performed, the block is read, and the transport is stopped at the next interblock gap.

### 2.8.7.3. Mode B Search

Used with the optional identifier-search feature, this mode expands the address-search capability to include a search for data characters within the text.

The tape is driven at high speed forward or backward to an address count of 20 preceding the address in the search information statement, and then the search continues at read speed to compare identifier characters in the search information statement with the first data characters in each block until a match is found. From 1 to 16 identifier characters may be supplied in the search information. When the search is completed, the block is read and the transport is stopped at the next interblock gap.

**NOTE:**

*If protected data is included in the identifier-search characters, the mode B search will not function.*

### 2.8.7.4. Mode C Search

This is a read-speed-only search for use with the optional identifier-search feature. No tape address is required in the search information, only the track number and the identifier characters. Although the mode C search is slower, it provides a search capability which can be used without building and saving an extensive tape directory.

When the mode C search is commanded, the cassette system initiates a forward read-speed search of the track indicated to compare the identifier characters with the first 1 to 16 data characters in the blocks. When a match is found, the block is read and the transport is stopped at the next interblock gap.

**NOTE:**

*If protected data is included in the identifier-search characters, the mode C search will not function.*

### 2.8.8. Backward-One-Block Tape Repositioning

This function, which can be commanded either online or offline, repositions the tape from its present interblock gap to the first preceding interblock gap or until a given amount of tape is moved without encountering another block or the beginning of tape. After the tape is repositioned, the transport stops and the cassette system is in read mode. Although the next block will not be automatically read, only a print command is required to perform the read operation. Any other operation will require a new selection.

### 2.8.9. Address Reporting

Tape-position addresses are displayed, as they are counted by the tachometer, on the face of the cassette unit. The current tape address is recorded on tape as the block is written, but this block address is not transferred in a read operation. However, with the use of a special processor control command (the report-address command), the block addresses written by the cassette system can be supplied to the processor.

If the address is requested following a write operation, the address reported is the one recorded in the heading of the block just written. Any subsequent report-address command after the same write operation will still give the same address. It takes a tape movement to change the address reported.

The report-address command would probably be used most often to build a tape directory. Offline, such a directory must be determined from the address indicators.

### 2.8.10. Beginning-of-Tape and End-of-Tape Detection

A small hole is punched in the magnetic-oxide-covered tape a measured distance from the clear leader at each end of the tape. These holes are sensed by the cassette system and are used to identify the beginning of tape (BOT) and the end of tape (EOT). The same hole is used to identify these locations regardless of which track is being used. The cassette system can sense the BOT and EOT holes at read, write, and search speeds.

When the BOT hole is sensed, the cassette system starts the measurement to load point (see 2.8.11).

In tape cassette system types 0866-00 and -01, the cassette system generates an EOT signal and then comes to an orderly halt when the EOT hole is sensed. However, when tape cassette system types 0866-02 and -03 are used with the UNISCOPE 200 terminal, a different method of detecting the EOT condition is necessary. Because of the increased data capacity of the UNISCOPE 200 terminal, if the EOT signal were generated at the normal hole location the remaining tape would not be long enough to completely record a full-screen data block. Therefore, tape cassette system types 0866-02 and -03 contain a strappable selection, inserted at the factory or strapped by the Sperry Univac customer engineer, which permits operation with either a UNISCOPE 100 terminal or a UNISCOPE 200 terminal. When strapped for UNISCOPE 200 terminal operation, the cassette system generates the EOT signal during a write operation at an address count of 6000 — approximately 250 counts ahead of the EOT hole. In a read operation, the address comparison is not utilized; the EOT hole is sensed and the EOT signal is generated the same as explained above for types 0866-00 and -01.

#### NOTE:

*If either hole is positioned directly over the sensor, the cassette system will interpret the tape position as clear leader, all operation will halt, and the tachometer will indicate 0000. An offline rewind will be required to reposition the tape so that the BOT or EOT hole is not over the sensor.*

### 2.8.11. Load Point

The cassette system begins reading and writing at a position, called the load point, which is a measured distance beyond the beginning-of-tape hole. The read load point address will be approximately 0027 ( $\pm 0002$ ); the write load point will be approximately 0031 ( $\pm 0002$ ).

## 3. Codes

### 3.1. GENERAL

This section describes the codes for all transactions between the processor and the UNISCOPE Display Terminal. These codes are derived from a 7-bit (plus parity bit) modified ASCII.

### 3.2. MESSAGE FORMATS

A message is a sequence of characters arranged for the purpose of conveying information from an originator to one or more destinations (or addresses). (Refer to 2.5.3.) The message elements are:

SOH

ADDRESS

SUPERVISORY SEQUENCE (if required)

STX (if the message contains data)

TEXT (if the message contains data)

ETX

BCC

### 3.3. MESSAGE TYPES

Essentially, three types of messages are used with the UNISCOPE terminal: messages to the terminal, messages from the terminal, and supervisory messages. Supervisory messages can be freestanding or combined with either of the other two types of messages.

#### 3.3.1. Supervisory Messages

A supervisory message is a sequence of one or more communications control characters, and possibly other characters, that performs a defined control function. The sequence of characters follows immediately after the last address character (DID) in the heading of the message. The communications control characters are shown in Table 3-1.

Table 3-1. Communications Control Characters

Function	Symbol to Terminal	Symbol from Terminal
Start of heading	SOH	SOH
Start of text	STX	STX
End of text	ETX	ETX
Status poll (enquiry)	ENQ	-
Reply request (enquiry)	-	DLE ENQ
Synchronizing character	SYN	SYN
Acknowledge	DLE 1	DLE 1
Retransmit request	DLE NAK	-
Break	DLE [	-
Resume	DLE ]	-
Busy (WABT)	-	DLE ?
Traffic response to status poll	-	DLE 0
THRU	-	DLE ;
Processor message waiting	BEL	-

### 3.3.2. Messages to Terminal

The terminal receives poll messages and text messages. Polls are either traffic polls or status polls. (DLE NAK can be thought of as a specialized poll.) Text messages to the terminal are data and the information needed to handle the data. BEL is a special text message.

#### 3.3.2.1. Polls

Polls from the processor are used to solicit traffic, status, or reply request message from the terminal. If an error is detected in a polling sequence, no response is sent from any terminal in the network. A terminal never transmits except when polled. All responses, except the no-traffic-without-acknowledge message, are identified with the address of the transmitting terminal.

##### 1. Traffic poll

The traffic poll solicits traffic from the remote terminal. Included in the terminal response will be any outstanding acknowledgment from the addressed terminal or (if a general poll) from any other terminal (recognizing the addressing characters) on the same multiplexer. A traffic poll may have either a specific or a general address (any station on a specific multiplexer). The traffic poll is distinguished by the absence of any other control sequence; it will have an acknowledge sequence if one is required. A message containing text cannot poll the remote terminal. Traffic poll formats are shown in Figure 3-1. The traffic poll may bring in a request-processor-message (BEL) response or a special function message or information from an auxiliary device.

##### 2. Status poll

The status poll solicits status information from the terminal. The response indicates if the terminal, or any terminal on a multiplexer if a general address is used, has any traffic. The traffic is not sent but is held for a traffic poll. The DLE 0 sequence in the response indicates that the terminal has traffic. The response includes any outstanding acknowledgment from the addressed terminal or from any other terminal on the same multiplexer. A status poll may have either a specific or a general address. The ENQ character is used to solicit status. Status poll formats are shown in Figure 3-1. The status poll may bring in a request-processor-message (BEL) response, a special function message, or information from an auxiliary device.

STATUS POLL WITHOUT ACKNOWLEDGE\*

SOH R/G S/G D/G ENQ ETX BCC

STATUS POLL WITH ACKNOWLEDGE\*

SOH R/G S/G D/G DLE 1 ENQ ETX BCC

TRAFFIC POLL WITHOUT ACKNOWLEDGE\*

SOH R/G S/G D/G ETX BCC

TRAFFIC POLL WITH ACKNOWLEDGE\*

SOH R/G S/G D/G DLE 1 ETX BCC

TEXT

SOH RID SID DID STX TEXT ETX BCC

PROCESSOR MESSAGE WAITING

SOH RID SID DID BEL STX ETX BCC

RETRANSMISSION REQUEST

SOH RID SID DID DLE NAK ETX BCC

BREAK - RESUME

...XXX DLE [ SOH COMPLETE FORMAT BCC DLE ] XXX...

URGENT PROCESSOR MESSAGE (OVERRIDE TERMINAL ACTIVITY)

SOH RID SID DID NUL NUL NUL STX NUL NUL NUL HT NUL NUL NUL STX TEXT ETX BCC

20 ms\*\*

10 ms\*\*

20 ms\*\*

(Terminates auxiliary  
interface activity)

(Suppresses transmit)

\*In these message formats, R/G indicates RID or GID, S/G indicates SID or GID, and D/G indicates DID or GID.

\*\*These times are for the UNISCOPE 100 terminal; they are doubled for the UNISCOPE 200 terminal.

Figure 3-1. Processor-to-Terminal Message Formats

### 3. Retransmission request — DLE NAK

The retransmission request from the processor causes the terminal to retransmit that which it sent last. It can be considered a negative acknowledgment. The retransmission request is sent in response to the reply request (DLE ENQ) from the terminal (3.3.3.2). Retransmission request formats are shown in Figure 3-1.

#### 3.3.2.2. Text

The text message is a sequence of characters which conveys the text and the information necessary to handle the text. It is from the processor or, in the response to a traffic poll, from a terminal. The text portion of the message begins with the STX character and ends with the ETX character:

STX TEXT ETX

Processor-to-terminal text message formats are shown in Figure 3-1.

#### 3.3.2.3. Break and Resume (Embedded Messages) — DLE [ and DLE ]

Messages may be embedded in other messages by delimiting the embedded message with the break (DLE [) and resume (DLE ]) characters. The embedded message must not be addressed to the same terminal as the broken message; that is, it must be addressed to a different terminal. The embedded message may not be broken (only one level of embedding). The DLE [ and DLE ] characters are not included in the block check of either the broken or the embedded message. The embedded message has an independent block check character (BCC). The DLE ] character initiates the resumption of the previous message. The break must occur between STX and ETX or DC1, DC2 if included; it must not occur between ETX and BCC. An example of embedded message format is shown in Figure 3-1.

### 3.3.3. Messages From Terminal

The terminal sends text messages or supervisory messages. Text messages are those containing an STX. Supervisory messages include no traffic, busy (WABT), acknowledge, THRU, request processor message, special function messages, and reply request.

#### 3.3.3.1. Text — Terminal to Processor

The text message is of the same nature as processor-to-terminal text (see 3.3.2.2). The terminal-to-processor text message is sent in response to a traffic poll. Formats of this message type are shown in Figure 3-2.

NO TRAFFIC WITHOUT ACKNOWLEDGE  
EOT EOT or EOT EOT ETX BCC

NO TRAFFIC WITH ACKNOWLEDGE\*  
SOH RID SID DID DLE 1 ETX BCC

REPLY REQUEST  
SOH RID SID DID DLE ENQ ETX BCC

WABT (BUSY)  
SOH RID SID DID DLE ? ETX BCC

REQUEST PROCESSOR MESSAGE (OR SPECIAL FUNCTION MESSAGE) WITHOUT ACKNOWLEDGE  
SOH RID SID DID BEL or 7 or G or W or g ETX BCC  
Special Function Messages

REQUEST PROCESSOR MESSAGE (OR SPECIAL FUNCTION MESSAGE) WITH ACKNOWLEDGE\*  
SOH RID SID DID DLE 1 BEL or 7 or G or W or g ETX BCC  
Special Function Messages

THRU WITHOUT ACKNOWLEDGE  
SOH RID SID DID DLE ; ETX BCC

THRU WITH ACKNOWLEDGE  
SOH RID SID DID DLE 1 DLE ; ETX BCC

RESPONSES TO STATUS POLL ONLY

TRAFFIC WITHOUT ACKNOWLEDGE  
SOH RID SID DID DLE 0 ETX BCC

TRAFFIC WITH ACKNOWLEDGE\*  
SOH RID SID DID DLE 1 DLE 0 ETX BCC

RESPONSES TO TRAFFIC POLL ONLY

TEXT WITHOUT ACKNOWLEDGE  
SOH RID SID DID STX TEXT ETX BCC

TEXT WITH ACKNOWLEDGE\*  
SOH RID SID DID DLE 1 STX TEXT ETX BCC

\*WABT (DLE ?) may be sent instead of an acknowledgment (DLE 1) by a terminal that has received text correctly but cannot accept more text immediately. (See 3.3.3.2.)

2186

Figure 3-2. Terminal-to-Processor Response Message Formats

### 3.3.3.2. Supervisory Messages – Terminal to Processor

Terminal-to-processor supervisory messages are communications sequences sent in response to various types of polls and reflect the condition of the terminal or its status.

#### 1. No Traffic

The no-traffic-without-acknowledge (EOT EOT or EOT EOT ETX BCC) response to a poll is sent by a terminal to indicate that the terminal has neither traffic nor an acknowledgment to send and is not missing a processor acknowledgment for text transmitted to the processor (reply request). If the terminal has no traffic but has an outstanding acknowledgment to send, the normal format with address and block parity check is sent. Formats for no-traffic responses with and without acknowledgment are shown in Figure 3-2. The long format no-traffic-without-acknowledge (EOT EOT ETX BCC) is required by the SPERRY UNIVAC Communications Terminal Module Controller (CTMC). Note that the no-traffic-without-acknowledge response does not include an address.

#### 2. WABT (Busy) – DLE ?

The WABT response by a UNISCOPE terminal indicates that the terminal's auxiliary interface is active under processor control. A WABT response is returned instead of an acknowledgment when a terminal receives error-free text and is transferring the data via the auxiliary interface. The treatment of this response is similar to that of an acknowledgment, with respect to format and inclusion in messages from other terminals on a common multiplexer. A specifically addressed poll to a terminal that is busy will always bring the WABT response. WABT responses following the first one do not have any acknowledgment connotation. This response will be sent only once in response to a general poll. WABT formats are shown in Figure 3-2.

#### 3. Reply Request – DLE ENQ

The reply request from a terminal solicits the acknowledgment for the last acknowledgeable transmission by that terminal to the processor. If a terminal that is owed an acknowledgment does not receive an acknowledgment within the next good poll that it recognizes, the terminal keyboard remains locked (if it was locked) and the terminal sends a reply request to the processor. This allows the processor to determine whether the message or the acknowledgment for the message was lost. The processor will send either a retransmission request or a poll with acknowledgment, as appropriate. The terminal will continue to respond to its polls with a reply request until the processor either acknowledges or requests a retransmission and subsequently acknowledges the retransmission. The reply request format is shown in Figure 3-2.

#### 4. THRU – DLE ;

The THRU message (DLE ;), sent from the terminal, indicates that the terminal has completed a processor-controlled auxiliary interface transfer. The THRU response normally is used only after a WABT response. However, slow polling rates and/or fast data transfers can create situations where THRU could be returned without WABT first being returned to the processor. In this case, an acknowledgment is included with the THRU. Formats for the THRU response are shown in Figure 3-2.

## 5. Special Function Codes

Special function messages and request processor messages (BEL) are not preceded by STX but are followed by ETX. These messages can be considered as supervisory messages. The control codes generated by the special function keys are:

<u>Key Label</u>	<u>ASCII Code</u>	<u>Octal Code</u>
F1	7	067
F2	G	107
F3	W	127
F4	g	147

The special function keys and MESSAGE WAITING key are functionally interlocked and only one of these keys can be used at a given time. When a special function key or MESSAGE WAITING key is pressed, additional pressing of any of these keys will not be effective until a correct acknowledge message is received by the UNISCOPE terminal in response to the initial key message. If the MESSAGE WAIT indicator was turned on by a BEL from the processor, the pressing of the MESSAGE WAITING key or a special function key and the sending of the processor-message-waiting request or special-function-key message extinguishes this light.

On the alternate form of the special function keys (key set B), key F4 is labeled HANG UP. Pressing this key causes a disconnect message (DLE EOT or DLE EOT ETX BCC) to be transmitted and the momentary dropping of the data-terminal-ready (DTR) signal. This function causes the modem to disconnect from the line.

### 3.3.4. Urgent Processor Message

The processor has the unconditional ability to force a message to be displayed, overriding any other terminal operation. If the operation being interrupted is a transmit-pending condition (or if a message is being composed on the screen), the processor sends the heading, an STX followed by several NUL characters (the number depends on the bit rate of the communication interface), an HT, several more NULs, another STX, and the processor message. If the operation being interrupted is an auxiliary interface function (printing, for example), the processor sends SOH, followed by a specific address (RID, SID, DID), several NUL characters, STX, and the processor message. The number of NULs in the worst case with both types of interruptions must fill 20 milliseconds for the UNISCOPE 100 terminal or 40 milliseconds for the UNISCOPE 200 terminal. An example of the combined format for urgent processor messages is shown in Figure 3-1. The specific DID causes a dump print operation, and the first group of NULs allows time to dump the auxiliary interface activity. The first STX locks the keyboard, and the NULs immediately following allow time to enter the transmit-pending state; this handles the case of the operator pressing the TRANSMIT key. The HT causes a dump transmit operation, and the succeeding NULs allow time to dump the transmit condition. The second STX unlocks the keyboard so that the operator can enter a message following the urgent processor message.

### 3.3.5. Interface Timing Considerations

Certain UNISCOPE terminal functions involve sequential access to all or part of the terminal storage. The timing requirements to perform these functions are shown in Table 3-2.

Once initiated, these functions blind the storage to any other access until they are completed. This poses no problem when keyboard initiation is used, since successive key depressions cannot occur rapidly enough to allow interference between functions. When the functions are initiated by way of the communications interface, it is normal procedure in UNISCOPE terminal programming to follow such functions with enough NUL codes, at the interface speed used, to provide time for the function to be completed before the next code requiring storage access arrives at the terminal.

The figures listed in Table 3-2 are fixed values, built into the terminal. The time fill provided by each NUL character is relative to the data transfer rate and must be calculated for each transmission speed. For example, during operation at 2000 bits per second (bps), 250 characters a second are transmitted and each character requires 4 milliseconds for transmission. At 2000 bps, therefore, each NUL character will time fill a 4-millisecond interval.

Not all of these functions require time fill characters at all times. If the time to complete a function is less than the time fill value of the NUL character, no time fill is required. For example, erasing to end of display when 300 character positions are involved takes 3 milliseconds. At 2000 bps, the time fill provided by the NUL character is 4 milliseconds; therefore, at 2000 bps, no time fill character is required for this operation.

Table 3-2. Interface Timing

Function and Control Code Sequence	Execution Time per Character	
	Terminal Without Protected Format	Terminal With Protected Format
Tab (HT)	10 <i>us</i>	10 <i>us</i>
Insert in line (ESC d)	10 <i>us</i>	10 <i>us</i>
Insert in display (ESC `D)	10 <i>us</i>	10 <i>us</i>
Insert line (ESC j)	10 <i>us</i>	10 <i>us</i>
Delete in line (ESC c)	10 <i>us</i>	20 <i>us</i>
Delete in display (ESC C)	10 <i>us</i>	20 <i>us</i>
Delete line (ESC k)	10 <i>us</i>	10 <i>us</i>
Erase to end of display (ESC a)	10 <i>us</i>	10 <i>us</i>
Erase to end of unprotected display* (ESC M)	—	10 <i>us</i>
Erase to end of line (ESC b)	10 <i>us</i>	10 <i>us</i>
Erase to end of field* (ESC K)	—	10 <i>us</i>
Forced text to UNISCOPE terminal in transmit condition (Figure 3-1)	20 ms**	20 ms**
Dump print (Figure 3-1)	20 ms**	20 ms**

\*Provided only in terminals with protected format capability.

\*\*Time for the complete function.

### 3.3.6. Control Codes and Sequences

The control codes and control code sequences used with the UNISCOPE terminal are shown in Table 3-3. Not all of the coding illustrated is necessary for programming the UNISCOPE terminal, but it is included for completeness of information. A definition of the codes is included in Table 3-3.

Table 3-3. Control Codes and Control Code Sequences

Function	Code to Terminal	Code From Terminal
Cursor positioning (3.5.1)	ESC VT Y X SI	--
SOE positioning (3.5.2.2)	--	ESC VT Y X NUL SI
Cursor return (new line)	CR	CR
Erase to end of display	ESC a (note 1)	--
Erase to end of line	ESC b	--
Delete in line	ESC c	--
Delete in display	ESC C	--
Insert in line	ESC d	--
Insert in display	ESC D	--
Scan left	ESC g	--
Scan right	ESC h	--
Scan down	ESC i	--
Scan up	ESC f	--
Character erase (space)	SP	SP
Tab	HT	--
Tab stop set	ESC HT	--
Tab stop	--	HT
Message waiting	BEL (note 2)	--
Cursor to home	ESC e	--
Insert line	ESC j	--
Delete line	ESC k	--
Erase field	ESC K (note 3)	--
Erase display	ESC M (note 3)	--
Request processor message	--	BEL (note 4)
Start blink marker	FS	FS (note 5)
End blink marker	GS	GS (note 5)
Lock keyboard	DC4 or ESC DC4	--
Print	DC2	--
Print transparent	ESC DC2	--
Start of entry (SOE)	RS	RS
Shift in (note 6)	SI	--
Shift out (note 6)	SO	--
Line feed (note 7)	LF	LF
Form feed (note 7)	FF	FF
Transmit (unprotected)	DC1	--
Transmit display	ESC DC1 (note 3)	--

## NOTES:

1. The ESC code is used to provide control sequences as defined in the proposed ANSI Standard for Code Extension Procedures for Information Interchange, published in the COMMUNICATIONS OF THE ACM, December 1968, "Control Set Extension: Use of ESC (ESCAPE)."
2. The processor message waiting symbol (BEL) must be placed between the address (RID, SID, DID) and the start of text (STX). No text is present. If it follows STX, it is placed in storage and does not light the processor message waiting light. It is displayed as a space in storage.
3. These code sequences are used only with protected format units equipped for full protected format control.
4. BEL sent to the processor in a nontext message (no STX) indicates that the MESSAGE WAITING key was pressed. If BEL appears in text, it was placed in the UNISCOPE terminal either by the processor or through the auxiliary interface and is to be interpreted by the user program.
5. Cannot be inserted from the keyboard; must be inserted via I/O.
6. These characters control the function of characters immediately following them, shifting meaning from (SO) or to (SI) the standard graphics code being used. They are used to mark the beginning (SO) and end (SI) of protected format fields.
7. The UNISCOPE terminal is transparent to these codes. They are displayed as spaces. LF and FF may be placed in the terminal by either the processor or the auxiliary interface. (The term "transparent," as used here, signifies that the UNISCOPE terminal is unaffected by the characters referred to, but passes them for use by other devices.)

### 3.4. ADDRESSING AND ROUTING

The address immediately follows the SOH character. Three characters form the address with a standard format of RID (remote identifier), SID (station identifier), and DID (device identifier). ASCII graphic characters (Figure 2-4, columns 2 through 7) are used for the address. An address is included in all messages except the no-traffic-without-acknowledge response from the terminal to the processor.

A specific address includes a RID, a SID, then either a DID or a GID (general identifier). A general address can contain a GID in any or all of the address positions (RID, SID, and DID). It is normally used in the SID address position.

When UNISCOPE 100 and UNISCOPE 200 terminals are mixed on the same communications line, address identifiers must be assigned that will distinguish the smaller-capacity UNISCOPE 100 terminal from the larger-capacity UNISCOPE 200 terminal.

#### 3.4.1. Remote Identifier – RID

The RID is used for the first level of addressing. In a system using terminal multiplexers, the RID is used to address a group of display terminals connected to a terminal multiplexer. A RID must be used whether a multiplexer is used or not. Each RID is selected from the 48 graphic characters, SP through O, in columns 2, 3, and 4 of the ASCII code chart (Figure 2-4). The first character, SP (space), is assigned a nonspecific function, called a general identifier (GID). If this GID is used in the RID position, all groups and any connected terminal will recognize it as the RID part of its address. The GID should not be used in the RID position when two or more primary terminal multiplexers (multiplexers in multidrop configurations, each representing a separate remote address) are connected to the communications line, since more than one terminal might respond with indeterminate results. Caution should be exercised whenever a GID is used with a multidrop configuration, whether the drops are single station or multistation. In this case, the identification must be handled by the SID character.

#### 3.4.2. Station Identifier – SID

The SID is used for the second level of addressing. The SID specifies a terminal connected to the communications line by appropriate facilities either directly or through a multiplexer. The SID is selected from the 32 graphic characters (P through o) of columns 5 and 6 of the chart (Figure 2-4). The first character, P, is the general identifier (GID) similar to SP, as described in 3.4.1. When used with a RID specifying a group of terminals (on a multiplexer), this GID in the SID position allows any terminal in that group to recognize its address through the SID. The processor can address all of the terminals of that group, and the multiplexer can resolve the conflict if more than one terminal has traffic. Required polling is minimized because one poll can query multiple terminals.

#### 3.4.3. Device Identifier – DID

The DID is used for the third level of addressing. The DID specifies a particular auxiliary interface activity at a UNISCOPE terminal, such as a printer. The DID is selected from column 7 of the chart (Figure 2-4). There are 16 coded characters from p through DEL (delete) in column 7 ( % is the displayable character for DEL); however, the codes q, r, and DEL are not used for this application, and the first code, p, is the GID character for this level. This GID allows the processor to address the station without selecting a device.

#### 3.4.4. General Identifier – GID

The use of the general identifier, GID, is explained in the preceding paragraphs; however, a summary may be useful.

The address has three character positions corresponding to the three levels of addressing. The UNISCOPE terminal responds to two levels of addressing, the RID and the SID, and the auxiliary interface responds to the third level of addressing, the DID. The terminal or group of associated terminals at the first level recognizes two types of characters in the RID position, either the GID (which is the SP character) or a character specifically assigned to the terminal or group of terminals (RID). Following this, the individual terminal recognizes two characters in the second level position, either the GID (in this case, P) or a specific character (SID). The auxiliary interface responds to specific codes in the third level position; the GID (in this case, p) permits the terminal to function without specifying auxiliary interface activity. The address of SP, P, p is recognized by all terminals in the system.

It is possible to assign the same RID to all terminals on one multiplexer (Figures 3-3, 3-4, and 3-5). A general poll in such a configuration would then consist of the specific RID or a general RID (if the line is single drop), a general station identifier (GID), and a general device identifier (GID).

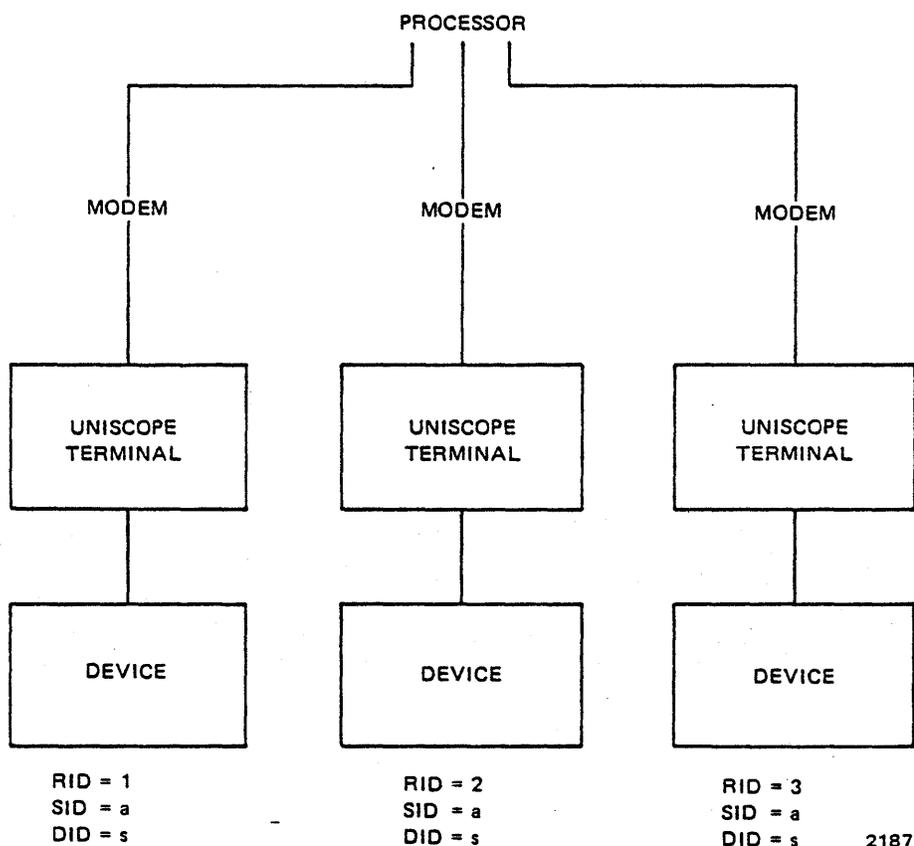
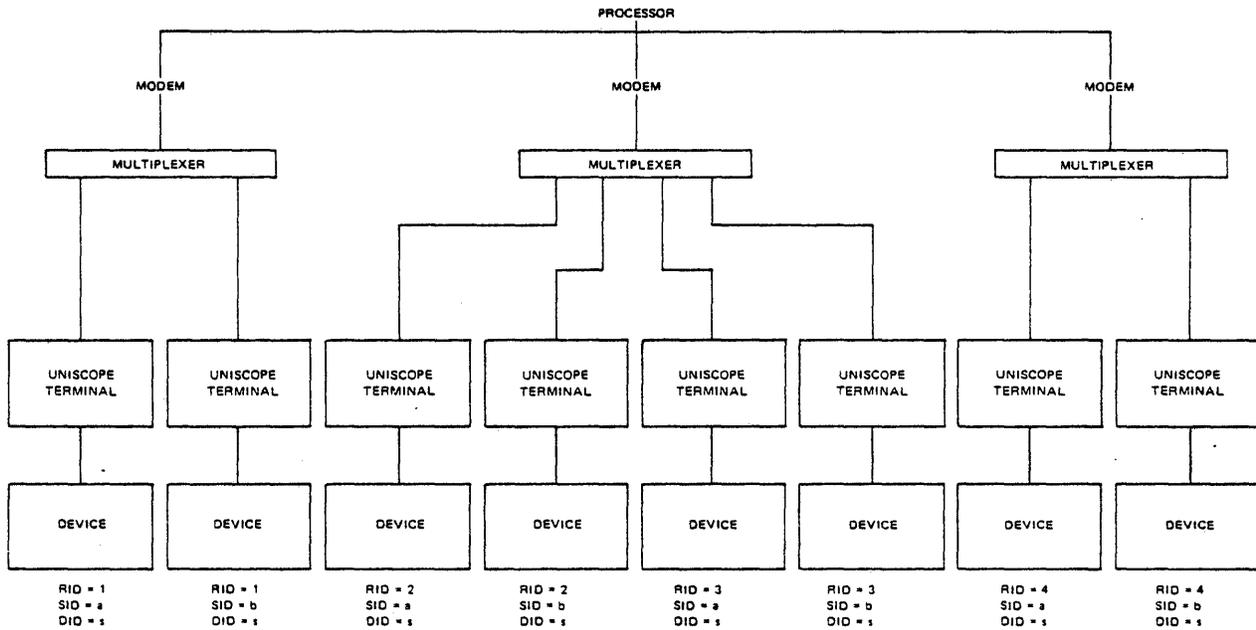


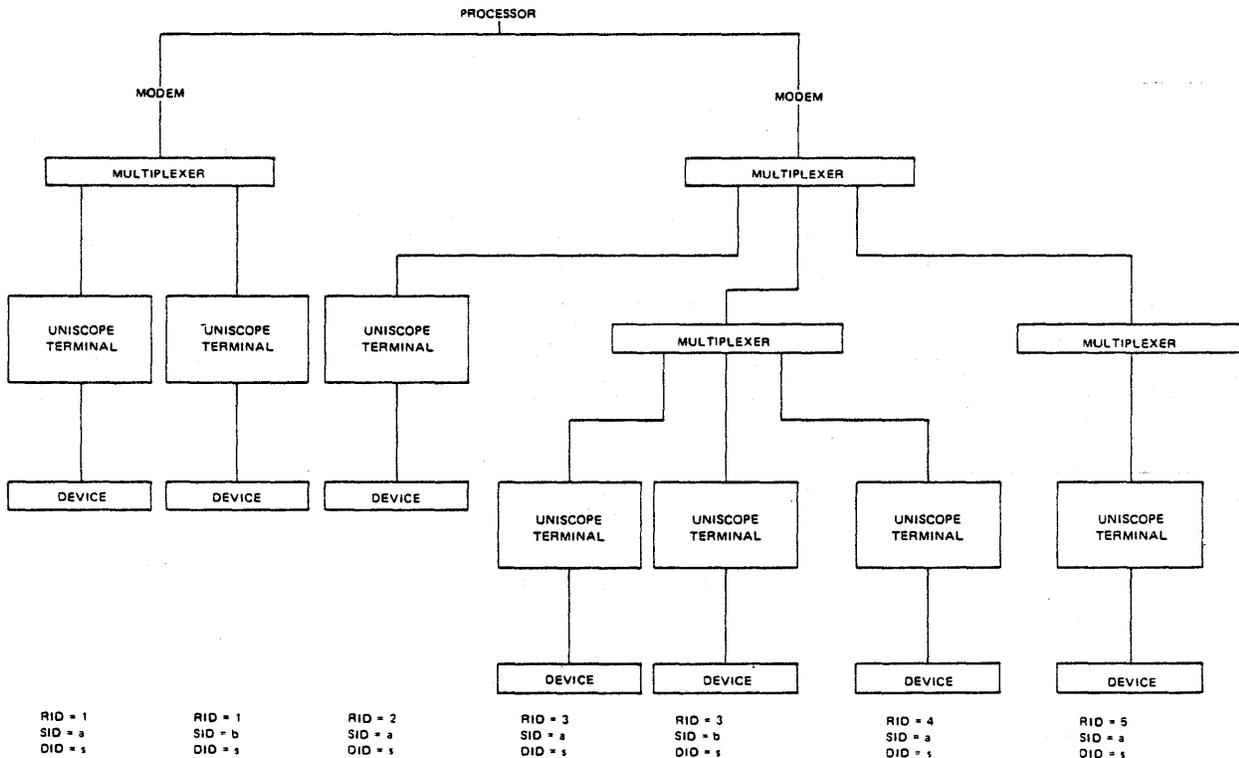
Figure 3-3. Example of Address Code Assignments for Single-Station Configurations





2190

Figure 3-6. Example of Address Code Assignments With One Multiplexer Divided Into More Than One Logical Multiplexer Function



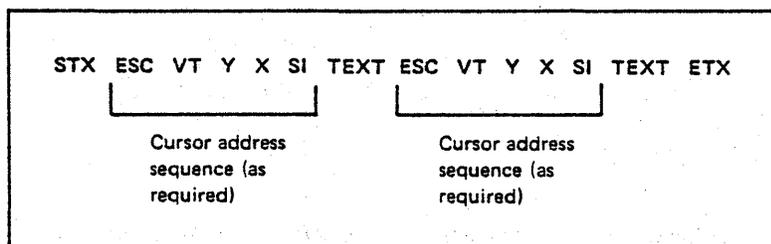
2191

Figure 3-7. Example of Address Code Assignments With Primary and Cascaded Multiplexers, One Primary/Cascaded Multiplexer Combination Being Divided Into More Than One Logical Multiplexer Function

### 3.5. TEXT FORMAT

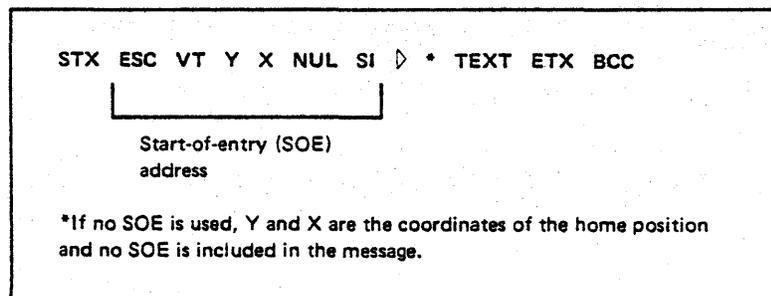
The formats of text between the processor and the UNISCOPE terminal are shown in Figures 3-8 and 3-9. In transmission from the processor to the UNISCOPE terminal, the text is placed on the screen of the UNISCOPE terminal starting at the position of the cursor. The cursor may be positioned by several different methods during transmission:

- The cursor address sequence positions the cursor to any specified point on the screen.
- The cursor-to-home sequence places the cursor in the upper left-hand corner of the screen.
- The cursor return and cursor move sequences position the cursor relative to its last position.
- The tabulation character moves the cursor in the same manner as the tabulation key of a typewriter (assuming tab stops are present).



2192

Figure 3-8. Text Format, Processor to UNISCOPE Terminal



2193

Figure 3-9. Text Format, UNISCOPE Terminal to Processor

#### 3.5.1. Cursor Address

The format for the cursor address in a text message from the processor to the UNISCOPE terminal is shown in Figure 3-8.

The ESC (escape) character determines the meaning of the character following it to permit extended control sequences. (See Table 3-3.) The VT character following ESC denotes that the next two characters are the cursor address; the first of the two identifies the horizontal line, and the second identifies the vertical column (according to the character assignments given in Figure 3-10).



The address coding is derived from columns 2 through 6 of the ASCII code chart (Figure 2-4). Columns 0 and 1 of the ASCII code chart are not used; these are for control. Column 7 is not used because no address greater than 80 is required with the UNISCOPE terminal. The exact code characters chosen are a function of the number of lines and columns in the particular UNISCOPE terminal being addressed and the desired position on the screen. The SI character is the ASCII character specified to end the sequence stated by the ESC VT code. The cursor address sequence may be placed anywhere in text after the STX character and before the ETX character. There is no limit on the number of times the cursor address sequence may be used in one message or on the positions that may be addressed.

### 3.5.2. Text

The text sent from the processor to the UNISCOPE terminal starts at the cursor position and moves to the right across the screen, superseding, character by character, any prior text displayed. A cursor return character (CR) in the text will cause the next character to appear at the beginning of the next line. If the end of a line is reached, the next character will appear at the beginning of the next line without the need for a cursor return character.

#### NOTE:

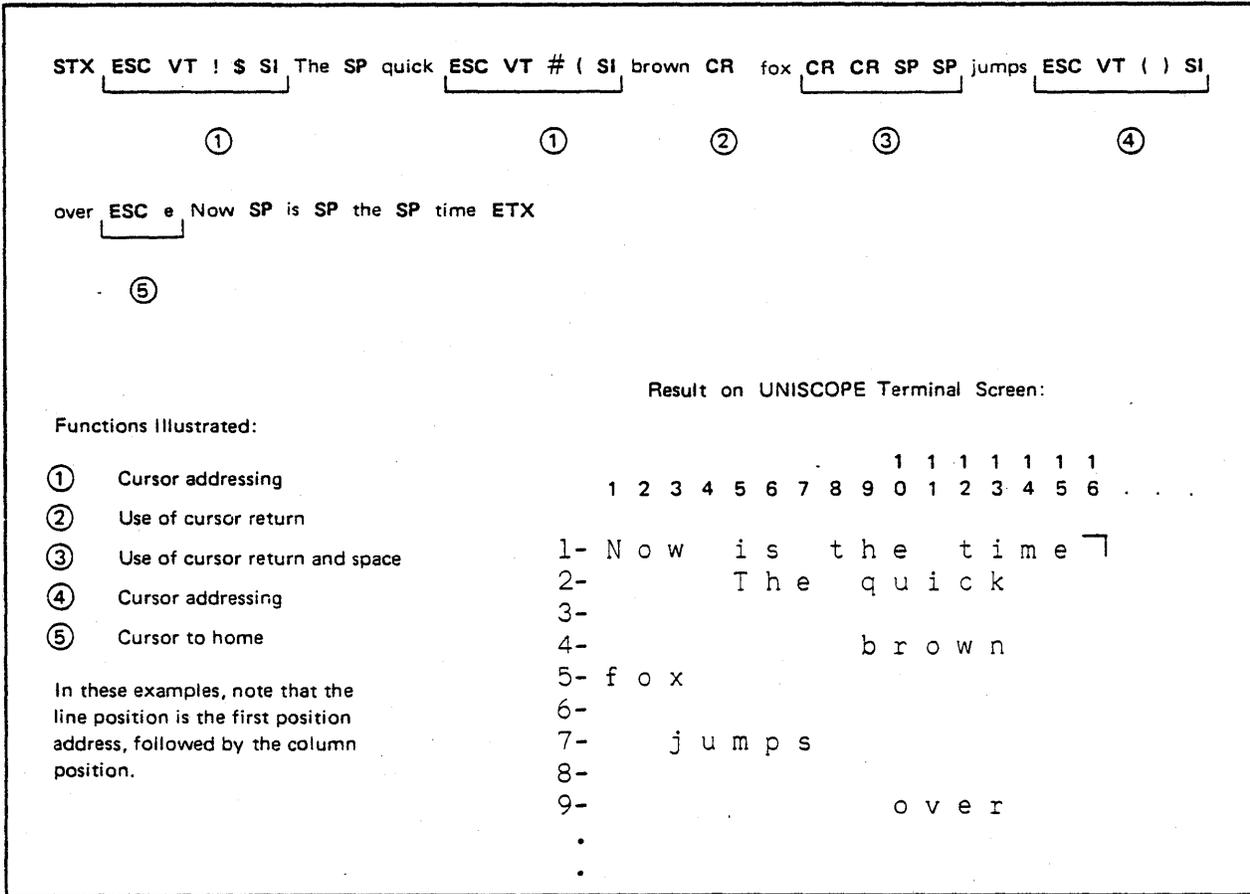
*If a CR is used even though the end of the line is reached, with the resultant automatic CR, a line skip will result.*

If a cursor return character is used before the end of a line, the rest of the line is unaffected. The erase-to-end-of-line sequence (ESC b) may be used to erase the rest of the line. The erase-to-end-of-display sequence (ESC a) may be used to clear the remainder of the screen following the end of the message, if desired.

#### 3.5.2.1. Data Format, Processor to UNISCOPE Terminal

An example of the data format used in transmission from the processor to the UNISCOPE terminal is shown in Figure 3-11. In the first part of this example, the phrase *The quick brown fox jumps over* is transmitted to the UNISCOPE terminal, but by use of the cursor addressing sequence and the cursor return, the data is placed on the screen as shown in the right of Figure 3-11. The initial cursor positioning sequence (ESC VT ! \$ SI) places the first letter (*T*) on the second line in the fifth position. The rest of the first two words (*The quick*) are placed on the screen in sequence. The second cursor positioning sequence (ESC VT # ( SI) places the first letter of the third word (*brown*) on the fourth line and in the ninth character position. The rest of the word follows in sequence. The next cursor positioning sequence is the cursor return (CR) at the end of the word which causes the first letter of the fourth word (*fox*) to be in the first character position of the next line. Then the two cursor returns followed by two spaces at the end of the fourth word cause the first letter of the fifth word (*jumps*) to be in the third position of the seventh line. The next cursor positioning sequence (ESC VT ( ) SI) causes the first letter of the last word in this phrase to be in the tenth character position of the ninth line.

Then, in the second part of the example, the phrase *Now is the time* is placed on the first line of the screen. This is done by positioning the cursor in the home position (ESC e). The phrase is then entered, one character at a time, from column 1, line 1 through column 15, line 1. The cursor remains at column 16, line 1. Since no erase sequences were used, the screen remains as before the transmission except for the ten words transmitted and the space between the words in sequence on a line and the spaces before the word on line 7. This is an example of the capability that allows the processor to change specific characters, words, phrases, sentences, or paragraphs on the screen without disturbing or repeating the data that is already on the screen.



2195

Figure 3-11. Data Format, Processor to UNISCOPE Terminal

### 3.5.2.2. Data Format, UNISCOPE Terminal to Processor

The format for data from the UNISCOPE terminal to the processor is shown in Figure 3-12. In this example, the TRANSMIT key is pressed while the text, *The Quick Brown Fox Jumps Over the*, shown at the top of the figure, is on the screen. The data that is transmitted is the portion between the cursor and the start-of-entry (  $\triangleright$  ) character nearest to the left of the cursor (in this example, the SOE between *Quick* and *Brown*). Immediately following STX is an escape sequence (ESC VT) giving the address of the SOE. The two characters following the ESC VT are the line and column positions (Figure 3-12) of the SOE character. The NUL character is inserted in the character location reserved for a future enhancement. The SI character is the ASCII symbol to end the address sequence. Immediately following the address sequence is the SOE character (RS); the first text character follows the SOE. The text continues to the last significant character on the line, in this case, *n*. A cursor return (CR) character is inserted, producing the effect of suppressing the nonsignificant spaces that follow.

The first character on the next line follows the cursor return. Spaces between words are included. A cursor return follows the *s*, and, again, nonsignificant spaces after the *s* in *Jumps* are suppressed. On the last line, the spaces are included between the last significant character, *e*, and the cursor; the nonsignificant characters following the cursor are suppressed. By transmitting the address of the SOE character at the beginning of the text, along with spaces out to the cursor, the processor program can accurately ascertain the location of the transmitted text on the face of the screen.

Since the processor can include SOE in its text message, it can insert the required SOE for the operator. With the SOE character positioned at the end of a question, the operator can simply type in the answer and press the TRANSMIT key without being concerned about either the SOE or the position of the data on the screen.

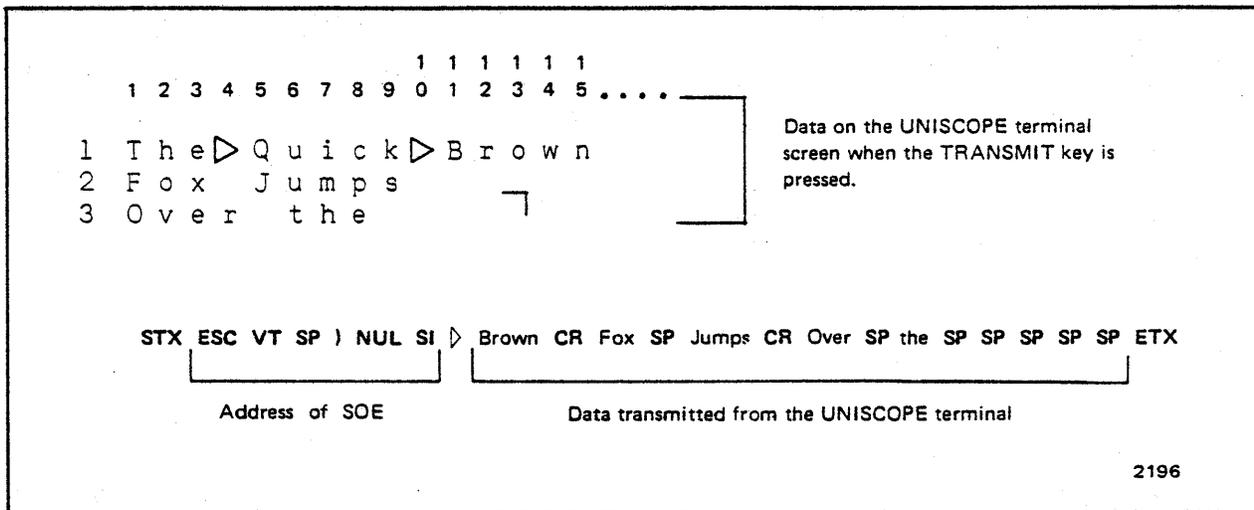


Figure 3-12. Data Format, UNISCOPE Terminal to Processor

### 3.5.3. Display Rolling (Scroll Effect)

The effect of the data rolling downward or upward across the face of the screen can be achieved by program use of the line-insert (ESC j) and line-delete (ESC k) edit functions.

The line-insert function causes the contents of the line containing the cursor and those lines below the cursor position to move down one line; the contents of the last line are removed from the screen. The result is a blank line inserted at the line containing the cursor. The cursor remains in the same position. With the transmission of data starting at the cursor position, data has been inserted. Repetition of this process causes the contents of the screen to move downward. Figure 3-13 illustrates this effect.

When the line-delete function is transmitted, the contents of the line containing the cursor are deleted and the lines below that line are all moved up to close the space; the bottom line is then blank. The cursor remains at the same position during this operation. Then, the cursor is repositioned at the bottom line and data is transmitted; in effect, the selected portion of the screen has been moved up one line. Repetition of this process causes the contents of the screen to move upward. Figure 3-14 illustrates this effect.

- 1 Begin with full screen. The cursor position is the top line for a full-screen roll-down effect:

```

1 But, in a larger sense, we cannot dedicate -- we cannot conse-
2 crate -- we cannot hallow -- this ground. The brave men, liv-
3 ing and dead, who struggled here, have consecrated it, far
4 above our poor power to add or detract. The world will little
5 note, nor long remember, what we say here, but it can never
6 forget what they did here. It is for us the living, rather, to
7 be dedicated here to the unfinished work which they who fought
8 here have thus far so nobly advanced. It is rather for us to
9 be here dedicated to the great task remaining before us, --
10 that from these honored dead we take increased devotion to that
11 cause for which they gave the last full measure of devotion --
12 that we here highly resolve that these dead shall not have died
13 in vain -- that this nation, under God, shall have a new birth
14 of freedom -- and that government of the people, by the people,
15 for the people, shall not perish from the earth.
16 -- A. Lincoln

```

- 2 ESC j at the top line inserts a blank line, moving the display down. The last line is moved off the screen:

Blank line inserted by ESC j

```

1
2
3 But, in a larger sense, we cannot dedicate -- we cannot conse-
4 crate -- we cannot hallow -- this ground. The brave men, liv-
5 ing and dead, who struggled here, have consecrated it, far
6 above our poor power to add or detract. The world will little
7 note, nor long remember, what we say here, but it can never
8 forget what they did here. It is for us the living, rather, to
9 be dedicated here to the unfinished work which they who fought
10 here have thus far so nobly advanced. It is rather for us to
11 be here dedicated to the great task remaining before us, --
12 that from these honored dead we take increased devotion to that
13 cause for which they gave the last full measure of devotion --
14 that we here highly resolve that these dead shall not have died
15 in vain -- that this nation, under God, shall have a new birth
16 of freedom -- and that government of the people, by the people,
for the people, shall not perish from the earth.
-- A. Lincoln

```

Bottom line removed from screen

- 3 Then the preceding line of information is transmitted into the blank line existing at the top of the screen:

New line inserted in blank space

```

1 It is altogether fitting and proper that we should do this.
2 But, in a larger sense, we cannot dedicate -- we cannot conse-
3 crate -- we cannot hallow -- this ground. The brave men, liv-
4 ing and dead, who struggled here, have consecrated it, far
5 above our poor power to add or detract. The world will little
6 note, nor long remember, what we say here, but it can never
7 forget what they did here. It is for us the living, rather, to
8 be dedicated here to the unfinished work which they who fought
9 here have thus far so nobly advanced. It is rather for us to
10 be here dedicated to the great task remaining before us, --
11 that from these honored dead we take increased devotion to that
12 cause for which they gave the last full measure of devotion --
13 that we here highly resolve that these dead shall not have died
14 in vain -- that this nation, under God, shall have a new birth
15 of freedom -- and that government of the people, by the people,
16 for the people, shall not perish from the earth.

```

- 4 This process is repeated as long as the roll-down command is in effect.

1 Begin with full screen. The cursor position is the top line for a full-screen roll-up effect:

1 Fourscore and seven years ago our fathers brought forth on this  
 2 continent a new nation, conceived in liberty, and dedicated to  
 3 the proposition that all men are created equal. Now we are  
 4 engaged in a great civil war, testing whether that nation, or  
 5 any nation so conceived and so dedicated, can long endure. We  
 6 are met on a great battlefield of that war. We have come to  
 7 dedicate a portion of that field, as a final resting place for  
 8 those who here gave their lives that that nation might live.  
 9 It is altogether fitting and proper that we should do this.  
 10 But, in a larger sense, we cannot dedicate -- we cannot conse-  
 11 crate -- we cannot hallow -- this ground. The brave men, liv-  
 12 ing and dead, who struggled here, have consecrated it, far  
 13 above our poor power to add or detract. The world will little  
 14 note, nor long remember, what we say here, but it can never  
 15 forget what they did here. It is for us the living, rather, to  
 16 be dedicated here to the unfinished work which they who fought

2 ESC k at the top line deletes that line:

Line deleted by ESC k

1 Fourscore and seven years ago our fathers brought forth on this  
 2 continent a new nation, conceived in liberty, and dedicated to  
 3 the proposition that all men are created equal. Now we are  
 4 engaged in a great civil war, testing whether that nation, or  
 5 any nation so conceived and so dedicated, can long endure. We  
 6 are met on a great battlefield of that war. We have come to  
 7 dedicate a portion of that field, as a final resting place for  
 8 those who here gave their lives that that nation might live.  
 9 It is altogether fitting and proper that we should do this.  
 10 But, in a larger sense, we cannot dedicate -- we cannot conse-  
 11 crate -- we cannot hallow -- this ground. The brave men, liv-  
 12 ing and dead, who struggled here, have consecrated it, far  
 13 above our poor power to add or detract. The world will little  
 14 note, nor long remember, what we say here, but it can never  
 15 forget what they did here. It is for us the living, rather, to  
 16 be dedicated here to the unfinished work which they who fought

3 The lines below the empty line are moved up one line, filling the empty space and making room at the bottom for another line:

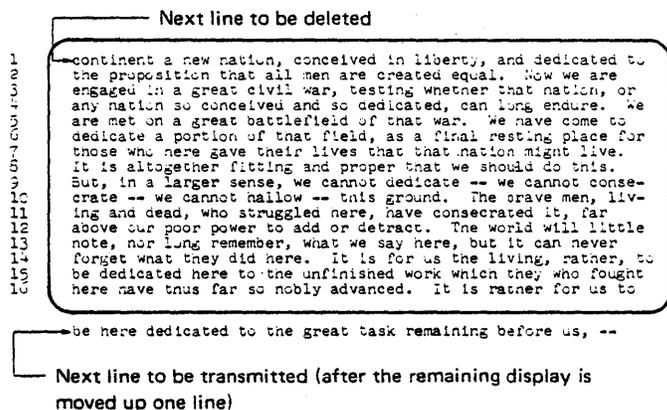
1 continent a new nation, conceived in liberty, and dedicated to  
 2 the proposition that all men are created equal. Now we are  
 3 engaged in a great civil war, testing whether that nation, or  
 4 any nation so conceived and so dedicated, can long endure. We  
 5 are met on a great battlefield of that war. We have come to  
 6 dedicate a portion of that field, as a final resting place for  
 7 those who here gave their lives that that nation might live.  
 8 It is altogether fitting and proper that we should do this.  
 9 But, in a larger sense, we cannot dedicate -- we cannot conse-  
 10 crate -- we cannot hallow -- this ground. The brave men, liv-  
 11 ing and dead, who struggled here, have consecrated it, far  
 12 above our poor power to add or detract. The world will little  
 13 note, nor long remember, what we say here, but it can never  
 14 forget what they did here. It is for us the living, rather, to  
 15 be dedicated here to the unfinished work which they who fought

here have thus far so nobly advanced. It is rather for us to

New line to be transmitted

Figure 3-14. Roll-Up Function (Part 1 of 2)

- 4 The cursor is moved to the bottom, and the next line of information is transmitted into the bottom space. Then the cursor is moved to the top line, and the top line is again deleted, the display is moved up one line, the cursor is moved to the bottom, a new bottom line is transmitted, and this process is repeated as long as the roll-up command is in effect:



2198

Figure 3-14. Roll Up Function (Part 2 of 2)

### 3.5.4. Tab Stop Setting

The character HT is placed in storage as the tab stop. The processor can insert tab stops in text transmitted to the UNISCOPE terminal. The sequence ESC HT is inserted at each point in the text where a tab stop is desired. Then, each time the operator presses the TAB key, the cursor moves sequentially to the next tab stop and stops one character position to the right of the tab stop. This makes it convenient for the operator to fill in blanks in forms. When the last tab stop is reached, the cursor will return to home position if the TAB key is pressed again. Also, the cursor will return home when the TAB key is pressed and no tab stop is present. If the tab stop is located in a protected field, pressing the TAB key will move the cursor to the first unprotected position to the right of the tab stop.

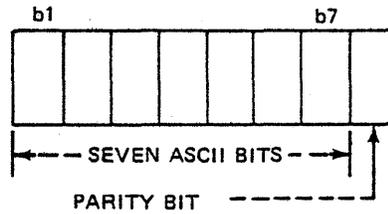
The operator can insert tab stops at any location by placing the cursor in that location and pressing the TAB SET key. Unlike a typewriter, the UNISCOPE terminal tab stops must be set at each position in each line. Thus, if a column arrangement is desired and tab stops are to be used to achieve it, the tab stop must be set at the desired position in each line. The tab stops become part of the data transmitted to the processor and must be reset for each new screen format. Tab stops are erased by overwriting.

### 3.5.5. Character Structure and Bit Sequencing

The character structure conforms to *ANSI X3.16, 1966, Character Structure and Character Parity Sense for Serial-by-Bit Data Communications in the USA Standard Code for Information Interchange*. The bit sequencing conforms to *ANSI X3.15, Bit Sequencing of the USA Standard Code for Information Interchange in Serial-by-Bit Data Transmission*. The character structure and bit sequencing applicable to the UNISCOPE terminal are summarized for synchronous and asynchronous transmission in the following paragraphs.

### 3.5.5.1. Synchronous Transmission

The character structure for synchronous data communications consists of eight bits; there are seven ASCII character bits plus one character parity bit (Figure 3-15).



**NOTE:**

The order of transmission is from left to right.

2199

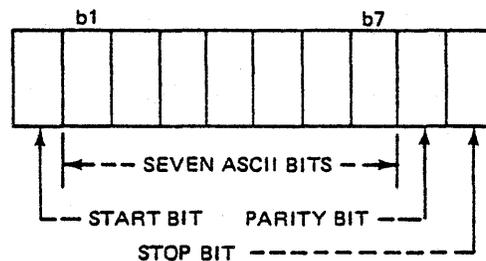
Figure 3-15. Character Structure, Synchronous Transmission

The bit sequence for an ASCII character is from the least significant bit (b1 first) to the most significant bit (b7) in terms of the ASCII nomenclature (*ANSI X3.4, Code for Information Interchange*) in ascending order.

When transmitting, the UNISCOPE terminal generates a parity bit and adds it to every 7-bit code transmitted. When receiving, the terminal checks the character parity. The character parity for synchronous transmission is odd; that is, there is an odd number of 1 bits per character.

### 3.5.5.2. Asynchronous Transmission

The character structure for asynchronous data communications consists of 10 signal elements having equal time intervals; one 0 (spacing) start element, seven ASCII bits, one character parity bit, and one 1 (marking) stop element. The intercharacter interval (the time interval between the end of a stop element and the beginning of the next start element) may be of any length and is of the same sense as the stop (marking) element, that is, 1. (See Figure 3-16.)



**NOTE:**

The order of transmission is from left to right.

2200

Figure 3-16. Character Structure, Asynchronous Transmission

The bit sequence for an ASCII character is from the least significant bit (b1 first) to the most significant bit (b7) in terms of the ASCII nomenclature (*ANSI X3.4, previously cited*) in ascending consecutive order.

When transmitting, the UNISCOPE terminal adds the start, parity, and stop bits to the seven ASCII bits. When receiving, the terminal uses the start and stop bits to establish bit and character timing, checks parity, and acts on the seven ASCII bits. The character parity for asynchronous transmission is even; that is, an even number of 1 (marking) bits per character.

### 3.5.6. Parity

Character parity conforms to *ANSI X3.16, 1966*, previously cited. Message parity conforms to *ANSI X3.3.4/212, Proposed American National Standard Data Communications Control Procedures for the American National Standard Code for Information Interchange*. The sense of character parity for synchronous data and communications is odd over the eight bits; that is, there is an odd number of 1 bits per character.

The sense of character parity for asynchronous data communications is even over the eight bits (seven ASCII bits and the character parity bit); that is, there is an even number of 1 bits per character.

The message parity character is the block check character (BCC) defined in *ANSI X3.3.4/212*, previously cited.

During transmission, the UNISCOPE terminal generates a BCC and transmits it to the processor as the last character of every message. (On synchronous systems, a pad character is added for communications timing.) During receiving, the UNISCOPE terminal accumulates and checks the BCC.

The BCC is generated by taking the binary sum independently (without carry) on each of the seven individual levels of the transmitted code (b1 to b7). If a sum is odd, a 1 bit is put into the corresponding position of the BCC. (The longitudinal parity is even.) See Table 3-4.

Table 3-4. BCC Parity Character Coding

MSG	b7	b6	b5	b4	b3	b2	b1	Character Parity	
								Sync	Async
SOH	0	0	0	0	0	0	1	0	1
S	0	1	1	0	1	0	1	1	0
h	1	1	0	1	0	0	0	0	1
p	1	1	1	0	0	0	0	0	1
STX	0	0	0	0	0	1	0	0	1
A	1	0	0	0	0	0	1	1	0
ETX	0	0	0	0	0	1	1	1	0
BCC	1	1	0	1	1	0	1	0	1

#### NOTES:

1. The BCC is the same (b1 through b7) for both synchronous and asynchronous transmission.
2. The SOH character is not included in the BCC calculation.

The character parity bit of the BCC character itself is the same sense as the character parity of the text characters; the parity is even for asynchronous transmission and odd for synchronous transmission.

The BCC calculation starts immediately following the SOH character. All characters that are transmitted (by the processor or the terminal) after SOH are included in the BCC calculation. This includes ETX, which signals that the next character is BCC. However, for processor-originated messages, it excludes the break and resume sequences and all characters between them. For the long format no-traffic-without-acknowledge response, both EOTs and the ETX are included in the accumulation.

For processor-to-terminal messages, the BCC must immediately follow the ETX character. SYN must not be transmitted between ETX and BCC.

### 3.5.7. Protected Format

Protected information is transmitted from the processor to the terminal with the SO and SI codes marking the beginning and ending of the protected data fields. An example of such a transmission is shown in Figure 3-17.

```
SOH RID SID DID STX ESC e SO NAME SI CR SO ACCOUNT SP NUMBER SI CR SO BALANCE SI CR
SO TRANSACTIONS SP SP DATE SI SP SO DEPOSIT SI . . . ETX BCC
```

2201

Figure 3-17. Example of Protected Information Transmitted from Processor to Terminal

The protected data fields are displayed on the terminal screen without any visible markers (Figure 3-18). However, the cursor and data cannot be placed in these fields by the operator; operator overwriting is not possible. (See 2.2.3.5 and 2.2.3.7.)

```

          1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 . . . .
1 NAME
2 ACCOUNT NUMBER
3 BALANCE
4 TRANSACTIONS DATE DE
.
.
.
2202
```

Figure 3-18. Example of Protected Information Displayed on Screen

The information composed on the terminal for transmission is placed in the unprotected fields, as appropriate. (See Figure 3-19.) Transmission of the unprotected fields occurs when the TRANSMIT or TRANSMIT UNPROT DISPL key is pressed. The protected fields are not transmitted to the processor but are marked by the SUB code (octal 032). (See Figure 3-20.) This is the usual procedure for using protected format. However, the entire display can be transmitted by use of the TRANSMIT DISPL key (on those units equipped with protected format keys). In this case, no distinction is made between the protected data and the unprotected data.

```

          1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 . . . .
1 NAME JOHN HENRY BROWN
2 ACCOUNT NUMBER 1229AB23872
3 BALANCE $1995.37
4 TRANSACTIONS DATE 19JUN72 DE
.
.
.
2203
```

Figure 3-19. Example of Unprotected Information Added to Protected Format Display

```
SOH RID SID DID STX ESC VT SP SP NUL SI SUB SP JOHN SP HENRY SP BROWN
CR SUB SP 1229AB23872 CR SUB SP $1995.37 CR SUB SP 19JUN72 SUB . . . ETX BCC
```

2204

Figure 3-20. Example of Unprotected Information Transmission from Terminal to Processor

Nonsignificant space suppression is performed within the transmitted unprotected fields as well as at the end of the lines. When more than one unprotected field occurs in a line, nonsignificant space suppression occurs more than once in that line.

### 3.6. CODE DEFINITIONS

The discussion of codes as related to the UNISCOPE terminal is covered in three areas:

1. communications control characters used to control the communications over the communications link;
2. text (or data) characters used to convey the actual message information; and
3. function codes used to control the UNISCOPE terminal.

Except where otherwise stated, the code used is *ANSI X3.4, 1968, Code for Information Interchange*. Octal codes for the control characters are given in Table 3-5; octal codes for the printable characters are given in Table 2-2; control characters that are not placed in storage are given in Table 2-1.

Table 3-5. Control Characters and Corresponding Octal Codes Transmitted

Control Character	Octal Code	Control Character	Octal Code
NUL	000	DLE	020
SOH	001	DC1	021
STX	002	DC2	022
ETX	003	DC3	023
EOT	004	DC4	024
ENQ	005	NAK	025
ACK	006	SYN	026
BEL	007	ETB	027
BS	010	CAN	030
HT	011	EM	031
LF	012	SUB	032
VT	013	ESC	033
FF	014	FS	034
CR	015	GS	035
SO	016	RS	036
SI	017	US	037

### 3.6.1. Standard Communications Control Characters

#### 3.6.1.1. SOH – Start of Heading

SOH is a communications control character used at the beginning of a sequence of characters that constitutes a machine-sensible address or routing information. This sequence is referred to as the heading. The heading is terminated by STX or by ETX depending on whether it is accompanied by text or by a supervisory sequence. (See 3.6.1.2 and 3.6.1.3.)

**Usage:**

SOH marks the start of a message heading.

#### 3.6.1.2. STX – Start of Text

STX is a communications control character which precedes a sequence of characters that is referred to as text. STX may be used to terminate a sequence of characters (heading) started by SOH.

**Usage:**

- STX marks the start of a message text.
- If a heading precedes the text, STX marks the end of the message heading.

#### 3.6.1.3. ETX – End of Text

ETX is a communications control character used to terminate a sequence of characters started with STX or SOH.

**Usage:**

- ETX marks the end of a message text.
- ETX may also mark the end of a heading.

#### 3.6.1.4. EOT – End of Transmission

EOT is a communications control character used to indicate no traffic without acknowledge in response to a poll. It is used in a redundant form (EOT EOT) to increase reliability.

**Usage:**

- Receipt of the EOT sequence by a processor, following a poll, indicates that the polled terminal (or group of terminals on a multiplexer) has no traffic, ACKs, or WABTs to send.
- The EOT sequence is never used to end a message from the terminal to the processor.
- The EOT sequence is never transmitted from a processor to a terminal.

### 3.6.1.5. ENQ – Enquiry

ENQ is a communications control character used to distinguish status and traffic polls.

ENQ is the basic character for the status poll.

Usage:

- ENQ is used to solicit status from a station.
- ENQ is not used to solicit retransmissions.

### 3.6.1.6. SYN – Synchronous Idle

SYN is a communications control character used by a synchronous transmission system to provide a signal from which synchronism may be achieved or retained.

Usage:

- SYN is used to achieve and maintain character synchronism in synchronous communications systems.
- SYN can be used as a communications time-fill character during periods in a transmission when no other characters are available to send.
- SYN may be transmitted as follows:
  - After a period in which no characters have been transmitted on a channel and prior to the transmission of any other character, at least four SYN characters must be transmitted.
  - Determination of synchronization, once achieved, and the recognition of character synchronization are the responsibility of the receiving station. No station is considered synchronized until two successive SYN characters have been received. All stations in a multipoint mode remain in synchronization when not sending if data is present on the receive line.
  - When the SYN character is used as a communications time-fill character during a transmission, SYN may be arbitrarily added at any point in the transmission except:
    1. in a control sequence following DLE;
    2. between ETX and the next following BCC (3.6.3.6); and
    3. between the address identifiers in a heading.
  - The SYN character is not to be used for time-fill or media-fill functions that are to be conveyed through the system. The receiving station deletes all SYN characters.

### 3.6.1.7. DLE – Data Link Escape

DLE is a communications control character that changes the meaning of a limited number of immediately following characters. It is used only to provide supplementary controls in data communications networks.

#### Usage:

Additional control functions are represented by an unbroken sequence of characters, the first character of which is always DLE.

### 3.6.2. Code-Extended Communications Control Characters

#### 3.6.2.1. WABT – Wait Before Transmit (Busy)

WABT is a communications control sequence sent by a remote station when an error-free block or message has been received but the station is temporarily unable to receive more traffic (busy). It is made up of the WABT character sequence: DLE ?

#### Usage:

WABT is transmitted by the terminal when the auxiliary interface is active because of a processor command.

#### 3.6.2.2. DNAK – Negative Acknowledgment

The DNAK is a communications control sequence sent by a processor to indicate a negative reply. It comprises the character sequence, DLE NAK.

#### Usage:

DNAK is transmitted by a processor as a retransmission request. It is used in response to a DENQ reply request from the terminal.

#### 3.6.2.3. DBR – Break (Before Embedded Segment)

DBR is a communications control sequence used to break a text message from the processor for the purpose of inserting another message to another station. It is represented by the character sequence, DLE [.

The DBR sequence is not included in the BCC of either message block (interrupted message or embedded message).

#### Usage:

- DBR identifies the characters following it as being in a new block.
- DBR suspends the calculation of the BCC of the first block.
- DBR causes the receiver of the first block to stand by.

### 3.6.2.4. DRE – Resume (After Embedded Segment)

DRE is a communications control sequence used to terminate the insertion begun by DBR. It is represented by the character sequence, DLE ] .

The DRE sequence is not included in the BCC of either message block (interrupted message or embedded message).

Usage:

- DRE marks the end of an embedded message.
- DRE causes the interrupted receiver to resume operation.
- DRE initiates the resumption of the calculation of the BCC for the original block.

### 3.6.2.5. DENQ – Reply Request

DENQ is a communications control sequence used by the terminal in data communications systems as a request for an acknowledgment. It comprises the character sequence, DLE ENQ.

## 3.6.3. Miscellaneous Control Characters or Sequences

The miscellaneous control characters are not necessarily communications controls, but they are related and are included here as a matter of record.

### 3.6.3.1. NUL – Null

The NUL is an all-zeroes character which may serve to accomplish time fill and media fill.

Usage:

The NUL character is used for time-fill or media-fill functions that are to be conveyed through the system.

### 3.6.3.2. SO – Shift Out

SO is a control character indicating that the code combinations that follow are not to be interpreted as a member of the character set of the standard (ASCII) code table until a shift in (SI) character is reached.

Usage:

SO is used to mark the beginning of a protected field.

### 3.6.3.3. SI – Shift In

SI is a control character indicating that the code combinations that follow are to be interpreted according to the standard (ASCII) code table.

**Usage:**

- SI is used to mark the end of a protected field.
- SI is used to mark the end of a cursor address sequence.
- SI is used to mark the end of an SOE address sequence.

**3.6.3.4. ESC – Escape**

ESC is a control character intended to provide code extension (supplementary characters) in general information interchange. The escape character itself is a prefix affecting the interpretation of a limited number of contiguously following ASCII characters.

**Usage:**

The combination of ESC and the characters following it is used to represent a control function not directly represented within the code.

**3.6.3.5. BCC – Block Check Character**

BCC is a character added at the end of a message to facilitate error detection as follows:

1. The BCC is generated by taking a binary sum independently (without carry) on each of the seven individual levels of the transmitted code (b1 to b7).
2. In each code level the number of 1 bits, including any in the BCC, is caused to be even. Thus, the sense of longitudinal parity is said to be even whether the transmission is synchronous or asynchronous.
3. The correct value of the character parity bit of the BCC itself is that which makes the sense of character parity the same as the text character.

**Usage:**

- Counting for the BCC is started immediately following the first appearance in a message or transmission block of either SOH or STX. STX may appear in a block starting with SOH; in that case, the count includes STX.
- All characters following the delimiting character (SOH or STX) are included in the summation except SYN characters and embedded messages with their delimiters, DBR and DRE. The summation includes the delimiting control character which signals that the next following character is the BCC.
- The BCC is transmitted as the next character following the transmission of each end-of-text (ETX) character.
- For the long format no-traffic-without-acknowledge response, the BCC includes both EOTs and the ETX (EOT EOT ETX BCC).

### 3.6.4. Data Characters

The data characters are the graphic symbols taken from *ANSI X3.4, 1968, Standard Code for Information Interchange*. The character set is shown in Figure 2-4. The 64-character option includes the characters shown in columns 2 through 5. The 96-character option includes the characters in columns 2 through 7.

### 3.6.5. Editing and Device Control Codes

Refer to Table 3-3 to determine which codes are sent from the processor to the terminal and which codes are sent from the terminal to the processor.

#### 3.6.5.1. CR – Cursor Return

The cursor return character causes the cursor on the terminal display to move to the beginning of the next line. The cursor return is not placed in storage. Sent from the UNISCOPE terminal to the processor, cursor return characters are inserted at the end of each line (after nonsignificant spaces are suppressed) in the data transmitted to the processor.

#### 3.6.5.2. ESC a – Erase to End of Display

The erase-to-end-of-display function causes erasure of all of the data from the current cursor position to the end of the display. The cursor remains in the same position throughout this function. For units provided with protected format control, the function is identified as erase-unprotected-display and functions the same as described but only on unprotected data.

#### 3.6.5.3. ESC b – Erase to End of Line

The erase-to-end-of-line function causes the erasure of the display from the current cursor position to the end of the line. The cursor remains in the same position throughout this function. For units with protected format, this function causes erasure of the characters from the cursor to the end of the unprotected field in which the cursor is positioned or to the end of the line on which the cursor is positioned, whichever occurs first. If, at the end of a message, the processor leaves the cursor positioned within a protected field, this key will not function until the cursor is moved into an unprotected field.

#### 3.6.5.4. ESC C – Delete in Display

**NOTE:**

*DELETE, as used in this control function, is equivalent to erase and close up.*

The delete-in-display function causes the character at the cursor position to be deleted. All of the following characters to the end of the display are moved left one position, and a blank is left at the end of the display. The cursor does not move from its position during this function. For units provided with protected format, the function operates as stated except that it operates only on unprotected data and is limited to one unprotected field rather than to the end of the screen. This permits shifting several lines within one unprotected field, but does not shift characters in any of the other unprotected fields in the same operation.

### 3.6.5.5. ESC c – Delete in Line

**NOTE:**

*DELETE, as used in this control function, is equivalent to erase and close up.*

The delete-in-line function causes the character at the cursor position to be deleted. The rest of the characters in the line are moved to the left to close the gap, and a blank is left at the end of that line. The cursor does not move from its position during this function. For units provided with protected format, the function operates as stated except that it operates only on unprotected data and only in one field. For example, if the field ends before the end of the line, the area shifted stops at the end of the unprotected area. Any following unprotected fields in the same line will not be shifted.

### 3.6.5.6. ESC D – Insert in Display

The insert-in-display function causes all of the characters beginning with the character under the cursor to be moved to the right one position; the last character on the display is lost if it is in the last position on the screen. A space is inserted at the cursor position. The cursor does not move from its position during this function. For units provided with protected format, the function is the same except the sequence is limited to one unprotected field. This permits the shifting of data within one unprotected field extending over more than one line, but does not shift data in any of the other unprotected fields in the same operation.

### 3.6.5.7. ESC d – Insert in Line

The insert-in-line function causes the character under the cursor and all of the characters to the right of the cursor on the same line to be moved to the right; the last character on the line is lost if it is in the last position on the line. A space is inserted at the cursor position. The cursor does not move from its position during this function. For units provided with protected format, the function is the same except the sequence is limited to one unprotected field on one line. The shifting stops at either the end of an unprotected field or at the end of the line, whichever occurs first.

### 3.6.5.8. ESC e – Cursor to Home

The cursor-to-home function moves the cursor to the home position (upper left-hand corner) of the screen (row 1, column 1).

### 3.6.5.9. ESC f – Scan Up

The scan-up function causes the cursor to move up one line. This function does not disturb the contents of storage.

### 3.6.5.10. ESC g – Scan Left

The scan-left function causes the cursor to move to the left one position. This function does not disturb the contents of storage.

### 3.6.5.11. ESC h – Scan Right

The scan-right function causes the cursor to move to the right one position. This function does not disturb the contents of storage.

### 3.6.5.12. ESC i – Scan Down

The scan-down function causes the cursor to move down one line. This function does not disturb the contents of storage.

### 3.6.5.13. SP – Space

The space character is a character used as a blank. The space character replaces the character in the terminal storage at the cursor position.

### 3.6.5.14. HT – Tab

The tab function causes the cursor to move to the next tab stop set in the storage of the terminal. The cursor will come to rest one character position to the right of the tab stop; overwriting will replace a tab stop with the character being written into that location.

For units provided with protected format, the tab function moves the cursor to the next HT character or to the first unprotected character location following the protected field containing the HT character. If all characters are protected, the cursor stops at home position.

### 3.6.5.15. ESC j – Insert Line

The insert-line function causes the line containing the cursor, and all following lines, to move down one line. A blank line results at the row the cursor is in, and the bottom line is deleted from the screen if it is in the last line position on the screen. The cursor remains in the same position throughout this part of the function.

### 3.6.5.16. ESC k – Delete Line

The delete-line function causes the line containing the cursor to be deleted, all of the lines below the line containing the cursor position to move up one line, and a blank line to appear at the bottom of the screen.

### 3.6.5.17. BEL (Processor) – Message Waiting

The message-waiting function is sent from the processor to the UNISCOPE terminal to indicate that the processor has an unsolicited message available for the operator. The function lights the MESSAGE WAIT indicator on the display panel and actuates the audible alarm. The actual meaning of this message and the indicator may vary from system to system according to local requirements. In many systems, it is used to permit the processor to present a message to the operator without destroying information being entered into the display unit by the operator. BEL has this meaning only when it is included in the message before STX. In this case, no text is present between STX and ETX. If BEL is included in the text of the message, it does not perform this function, but it is placed in storage and is displayed as a space on the screen.

### 3.6.5.18. BEL (Terminal) – Request Processor Message

When the operator presses the MESSAGE WAITING key, a message is generated for transmission to the processor (on a suitable poll); it contains the BEL character, which indicates to the processor that the MESSAGE WAITING key was pressed.

### 3.6.5.19. ESC HT – Tab Stop Set

The tab-stop-set function causes the display to place the HT character in storage to act as a tab stop (3.6.5.14). If text containing tab stops is returned to the processor, the tab stops (HT) will appear in the data. There is no restriction on the number of times the tab stop may be used.

### 3.6.5.20. FS – Start-Blink Marker

The start-blink marker (FS) is placed in storage; it causes the start-blink character to blink on the screen.

### 3.6.5.21. GS – End-Blink Marker

The end-blink marker (GS) is placed in storage; it causes the end-blink character to blink on the screen.

### 3.6.5.22. DC4 or ESC DC4 – Lock Keyboard

When the TRANSMIT key is pressed, the keyboard becomes locked (disabled). The keyboard is also locked when the start-of-text character (STX) is received from the processor. The keyboard is automatically unlocked at the end of the message (ETX followed by a satisfactory BCC) unless the lock keyboard character (DC4 or ESC DC4) is included at the end of the text. (Refer to 2.7.) This gives the programmer the ability to lock the keyboard until the next processor message.

### 3.6.5.23. DC2 – Print (Initiate Auxiliary Interface Function)

The DC2 code initiates a data transaction between the terminal storage and the auxiliary interface for either an input or an output device, whichever is selected. The format for the exchange is similar to that for exchanges via the communications channel; that is, the area to be transmitted is defined by the SOE character and the cursor, with suppression of nonsignificant spaces and the automatic insertion of cursor returns. The data exchanged between terminal storage and auxiliary interface appears in the screen format. Both protected and unprotected data are transferred under control of the DC2 code. There is no distinction made between protected and unprotected data.

### 3.6.5.24. ESC DC2 – Print Transparent (Initiate Auxiliary Interface Function)

The ESC DC2 code initiates auxiliary interface activity between the UNISCOPE terminal storage and the auxiliary interface in the same manner as the DC2 code. The area transmitted is the same as in the DC2 function, but the cursor return characters normally inserted by the logic of the UNISCOPE terminal are not transmitted to the auxiliary interface. This makes the line length of the device on the auxiliary interface independent of the line length of the particular UNISCOPE terminal in use. However, nonsignificant space suppression still occurs.

### 3.6.5.25. DC1 – Transmit, Transmit Unprotected Display

The transmit code may be transmitted from the processor to the UNISCOPE terminal. This function permits only that data within the unprotected areas on the screen to be transmitted to the processor. The SOE symbol is the only exception since it may be positioned in a protected area. The area transmitted is defined as the unprotected area between the SOE symbol nearest to the left of the cursor, and the cursor. In the transmission of data, each time a protected area is reached, the SUB code (octal 032) is inserted as a marker to indicate the omission of protected data. Nonsignificant space suppression is performed from the end of an unprotected field as well as from the end of a line; if there is more than one unprotected field on a line, nonsignificant space suppression occurs more than once. This allows nonsignificant space suppression within fields as well as on lines.

### 3.6.5.26. RS – Start of Entry (SOE)

The start-of-entry character (>) defines the beginning of the area to be transmitted to the processor or to the auxiliary interface device. If no SOE is present on the screen, the first row, first column position (home position) serves as the starting point. More than one SOE character may be on the screen (in storage) at one time. The SOE character nearest to the left of the cursor defines the beginning of the area that will actually be transmitted or transferred. The ability to have multiple SOE characters on the screen eliminates, for the programmer, the problem of removing these characters in instances where successive transmissions remain on the screen. It should be remembered that the cursor – and, thus, the text – will wrap around the display. Also, the programmer can place multiple SOE characters on the screen from the processor in order to aid the formatting of the next text to be transmitted. This might be a multiple choice by the operator, where the operator in positioning the cursor takes some action including a TRANSMIT key depression which would send back a controlled area. It might be used in a situation where the program sends repeated questions or answers, with a response requested from the operator by including the SOE character at the end of the message transmitted by the program. The operator can enter his response (question or answer, as appropriate) beginning at the position of the cursor and then press the TRANSMIT key at the end of this response without giving his attentions to the SOE character or to its transmitted location. This is useful when the operator is not well trained and where program control of the area to be transmitted is intended.

### 3.6.5.27. LF, FF – Line Feed and Form Feed

The ASCII line-feed (LF) and form-feed (FF) functions are intended for use by the UNISCOPE terminal for the auxiliary interface. The UNISCOPE terminal itself is transparent (passive) to these characters; they may be placed in storage by either the communications interface or the auxiliary interface and they appear as spaces on the UNISCOPE terminal.

### 3.6.5.28. ESC DC1 – Transmit Display

Used only with specially equipped protected format units, the transmit display (ESC DC1) code sequence transmits both protected and unprotected data to the processor with the protected fields not marked or identified.

### 3.6.5.29. ESC M – Erase Display

The erase display (ESC M) code sequence causes erasure of all data (protected and unprotected) from the cursor to the end of the display. The cursor does not move. This function is used on all UNISCOPE terminals except UNISCOPE 100 terminals with serial numbers below 6000.

### 3.6.5.30. ESC K – Erase Field

The erase field (ESC K) code sequence causes erasure of all unprotected data from the cursor to the end of the unprotected field within which the cursor is positioned (or end of display if it occurs first). If, at the end of a transmission, the cursor remains positioned within a protected field, this code sequence and the associated key will not function. The cursor must be moved to an unprotected field for this code sequence to again function. This function is used on all UNISCOPE terminals except UNISCOPE 100 terminals with serial numbers below 6000.

## 4. Tape Cassette System Programmer Information and Considerations

The communications control procedures for the SPERRY UNIVAC Model 610 Tape Cassette System are in accordance with those for the UNISCOPE terminal. It is assumed that the programmer is familiar with the UNISCOPE terminal information contained in this manual, and particularly with the discussion of the auxiliary interface. Refer to Section 2 for a functional description of cassette system operations.

### NOTE:

*In the following discussion, when the terms "cassette system" or "tape cassette system" are used without a type number, the information pertains to all types of cassette systems. The specific type numbers (types 0866-00 and -01, or types 0866-02 and -03) are used when applicable. Differences between the two types are covered in 2.8.*

For brevity in the message sequences illustrated in this section, the synchronizing (SYN) characters that precede each message in synchronous operation are not shown. On a synchronous line, of course, the synchronizing characters would be included. Also, all sequences are shown idealized; that is, the screen is assumed to be clear so that no erase-screen sequences are required, and it is assumed that there are no busy conditions.

### 4.1. SELECTION AND STATUS REPORTING

Each cassette system is designated by four device identifiers (DIDs), one to select the read function and one to select the write function for each transport. The DIDs are specified by the user and are strapped in at the user site.

#### 4.1.1. Selection Methods

Every processor-commanded cassette-system function (write, read, search, report address, and backward one block) is initiated with a selection message. The message may be either a poll (slow selection) or a text message (fast selection) containing a write or read DID for the desired transport.

A slow selection consists solely of a DID selection, as follows:

SOH RID SID DID ETX BCC

A fast selection consists of a DID selection plus a command, such as the following message containing a read command (assuming a read DID):

SOH RID SID DID STX DC2 ETX BCC

### 4.1.2. Status Reporting

When the selection message reaches the cassette system, the device deselects and the supplied DID is compared with the four cassette system DIDs. If there is a match, the cassette system selects and reports its condition to the auxiliary interface by means of a 4-bit status code which the UNISCOPE terminal translates to a device-status response and sends to the processor. The status codes and corresponding UNISCOPE terminal responses to the processor are listed in Table 4-1.

Table 4-1. Tape Cassette System Status Codes

Response Number	Mode	Bit Configuration (cassette system to auxiliary interface)				Description	UNISCOPE Terminal to Processor
		Bit 4	Bit 3	Bit 2	Bit 1		
1	Write	X*	0	1	1	Device ready to proceed	DLE >
2	Write	0	0	1	0	EOT	DLE <
3	Write	0	0	0	1	Data error**	DLE :
4	Write	0	0	0	0	Device unable to proceed/no response	DLE =
5	Read	X*	1	1	1	Device ready to proceed	DLE >
6	Read	1	1	1	0	EOT	DLE <
7	Read	0	1	0	1	Data error***	DLE :
8	Read	0	0	0	0	Device unable to proceed/no response	DLE =

\*Bit ignored by auxiliary interface

\*\*Timing errors in write operations

\*\*\*Includes both parity and timing errors

### 4.1.3. Status Code Interpretation

If the cassette system responds to the selection message with a device-ready-to-proceed code (1 or 5, Table 4-1), the processor may then send a text message containing either a DID or a GID (general device identifier).

**NOTE:**

*If a selection is followed by a text message with a DID, and the selection was made when the cassette was on clear leader, the message command will not be executed. Instead, the cassette system will deselect on reaching read or write load point.*

If the status response is an end-of-tape (EOT) code (2 or 6, Table 4-1), programming procedures may be accomplished as explained in 4.7.

If the status response is a data-error code (3 or 7, Table 4-1), the error reporting and recovery procedures described in 4.8 will occur.

If the status response is a device-unable-to-proceed/no-response code (4 or 8, Table 4-1), the cassette system will deselect (if already selected) and the commanded operation will be cleared. Possible reasons for this status response are:

- Cassette system power is not on.
- The cassette system is in an interlock condition because a cassette was not inserted properly or the transport door was not closed.
- A write function was commanded on a write-protected cassette.
- Problems associated with beginning-of-tape or end-of-tape procedures. (See 4.7.)
- A new selection was attempted while the cassette system was reading or writing a block. (See 4.8.3.)
- A new selection was attempted while a search or backward-one-block operation was in progress.
- A new write operation was attempted before the cassette system had completed the previous block termination sequence. (See 4.2.2.)

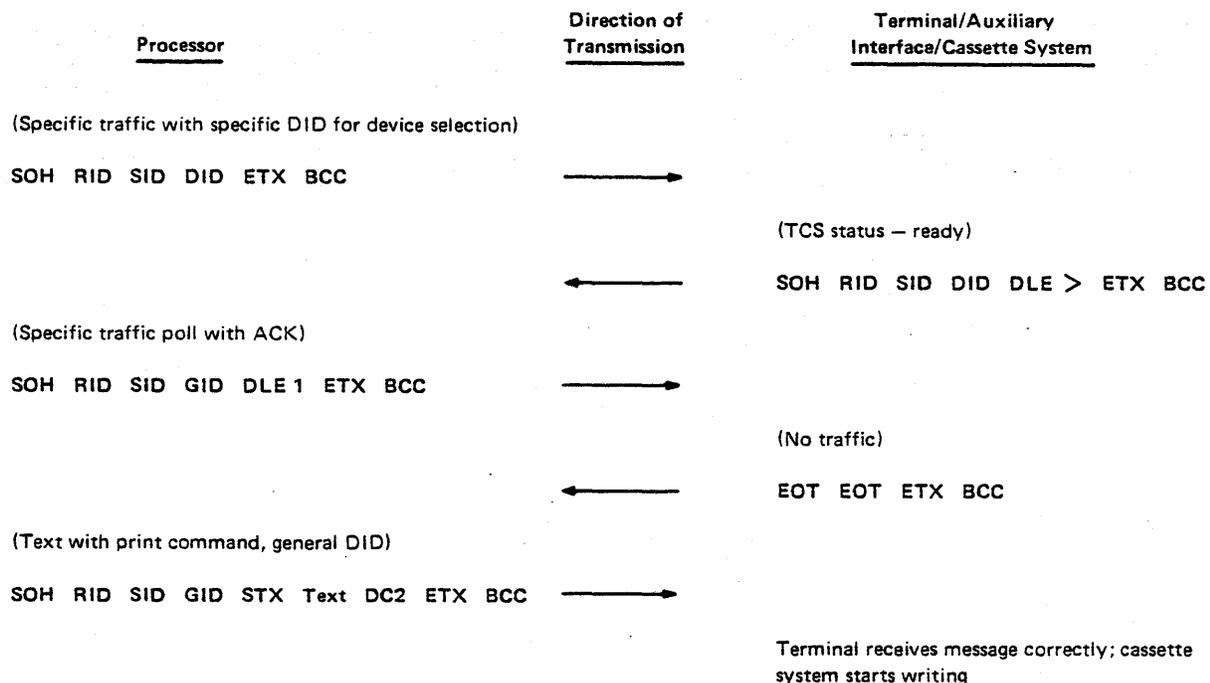
## 4.2. ONLINE WRITE

Data displayed on the terminal screen is transferred through the auxiliary interface to the cassette system with one print command and is recorded as one block. After the initial write DID selection, a specific DID is not required to write succeeding blocks of data.

### 4.2.1. Method

The following examples illustrate typical write sequences, using both the slow-selection and fast-selection methods. The DID is the write DID for the transport to be used in the operation.

#### Slow selection:



<u>Processor</u>	<u>Direction of Transmission</u>	<u>Terminal/Auxiliary Interface/Cassette System</u>
(Poll)		
SOH RID SID GID ETX BCC	→	(Busy, implied ACK)
	←	SOH RID SID DID DLE? ETX BCC
(Poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	(THRU)
	←	SOH RIQ SID DID DLE; ETX BCC
(Poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	(No traffic)
	←	EOT EOT ETX BCC
<b>Fast selection:</b>		
(Selection with text and print command)		
SOH RID SID DID STX Text DC2 ETX BCC	→	
(Poll)		
SOH RID SID GID ETX BCC	→	(Busy, implied ACK)
	←	SOH RID SID DID DLE? ETX BCC
(Poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	(THRU)
	←	SOH RID SID DID DLE; ETX BCC
(Poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	(No traffic)
	←	EOT EOT ETX BCC

### 4.2.2. Precautions

1. In certain cases, it may be useful to write a 1-character dummy block before the actual data block to act as a marker should a backward-one-block command (4.5.1) be performed in that area. Specifically, it is recommended that the dummy block be written as the first block after write load point and as the first block after a large gap between blocks of data (such as a section of tape that is deliberately left blank instead of using the automatic interblock spacing, which is uniformly measured). The backward-one-block command causes the tape to reposition back two blocks and then move forward one block and stop in the interblock gap between the two blocks. The dummy block will act as a marker in the two cases mentioned to maintain proper tape position.
2. At the end of each write operation, a block termination sequence (2.8.5.1) is placed on the tape. The terminal reports a THRU condition to the processor as soon as data transfer to the cassette system is completed. However, at this time the cassette system has not completed the block termination sequence, which requires approximately 160 milliseconds. If a specific selection is attempted before the cassette system has completed the block termination sequence, a device-unable-to-proceed/no-response status (4 or 8, Table 4-1) will be returned, and the cassette system will be deselected.

#### NOTE:

*The following precaution is not applicable to cassette systems used with a UNISCOPE 200 terminal or with models of the UNISCOPE 100 terminal containing auxiliary interface feature 1247-01. This auxiliary interface causes the cursor to back up one space on receipt of an end-of-text (ETX) signal from the input device. Therefore, the cursor wraparound condition described in the following paragraph is eliminated, permitting immediate copy or transfer of data to a processor without repositioning the cursor.*

3. When the UNISCOPE terminal writes a character, a line, or a full screen of data, the cursor is moved to the position immediately following the last character in preparation for writing the next character. This sequential positioning function is the same whether the data comes from a processor or from a tape cassette system. If the cursor was in the last screen position when a tape data block was written, the cursor will not be left in the last screen position when the data is read back into the terminal, but will be sequenced to the next position — the home position on the screen. (In other words, the cursor wraps around the screen to the home position.) If this data is then sent to another device, such as a printer, or transferred to a processor, only the character in the home position will be sent. To avoid this condition, which requires repositioning the cursor in a separate operation, it is recommended that two or three spaces be left between the cursor and the last position on the screen.

### 4.3. ONLINE READ

Each block of data on tape is transferred through the auxiliary interface to the UNISCOPE terminal with a print command and is transmitted from the terminal to the processor with a transmit command. After the initial read DID selection, a specific DID is not required to read succeeding blocks of data.

The cassette system generates an erase-screen sequence (ESC e ESC a or ESC e ESC M, depending on how the unit is strapped) if the cursor is not placed over the start-of-entry (SOE) symbol when a read operation is commanded.

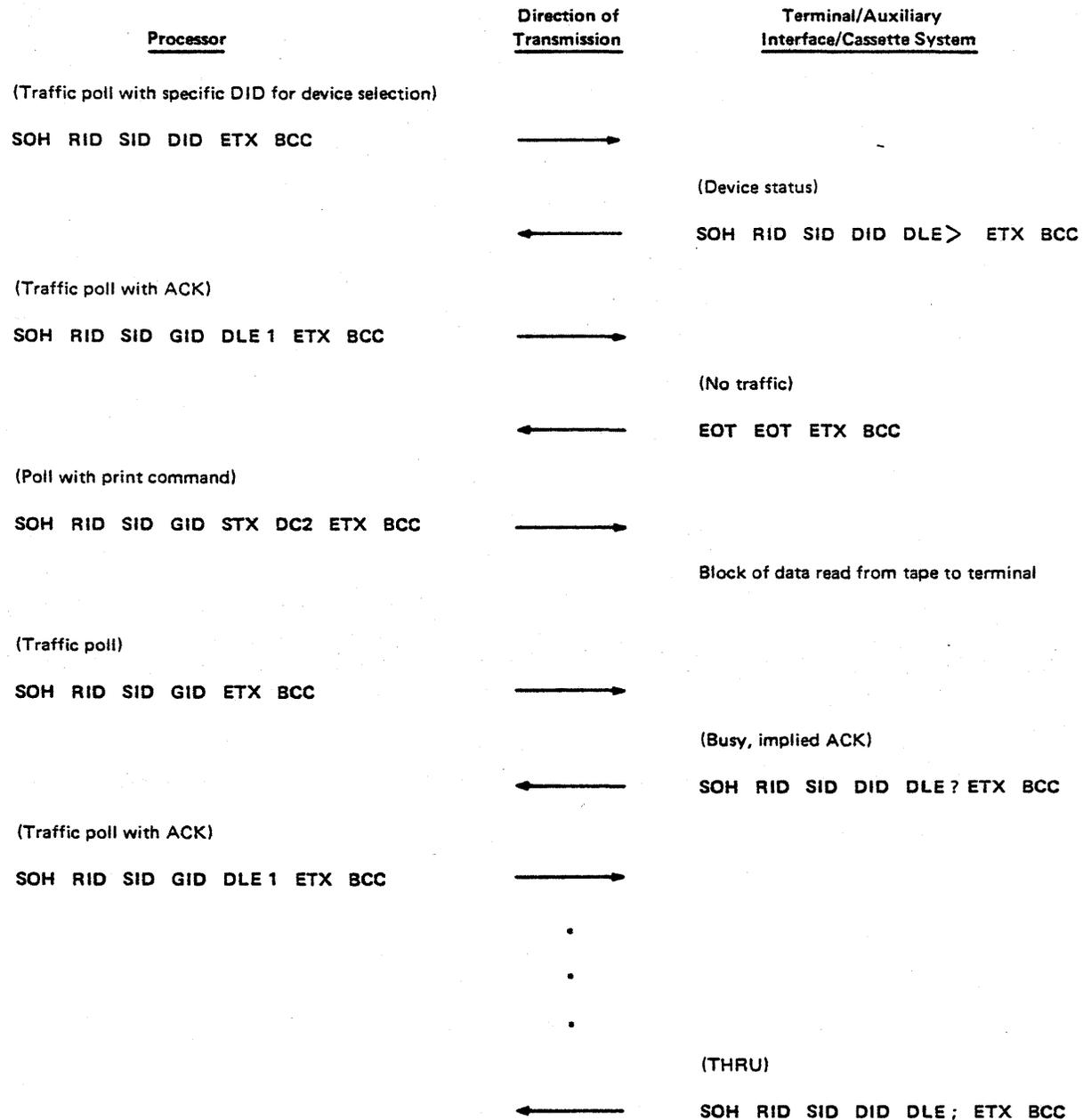
All heading and trailing characters are stripped from the block by the cassette system before the data passes through the auxiliary interface.

### 4.3.1. Method

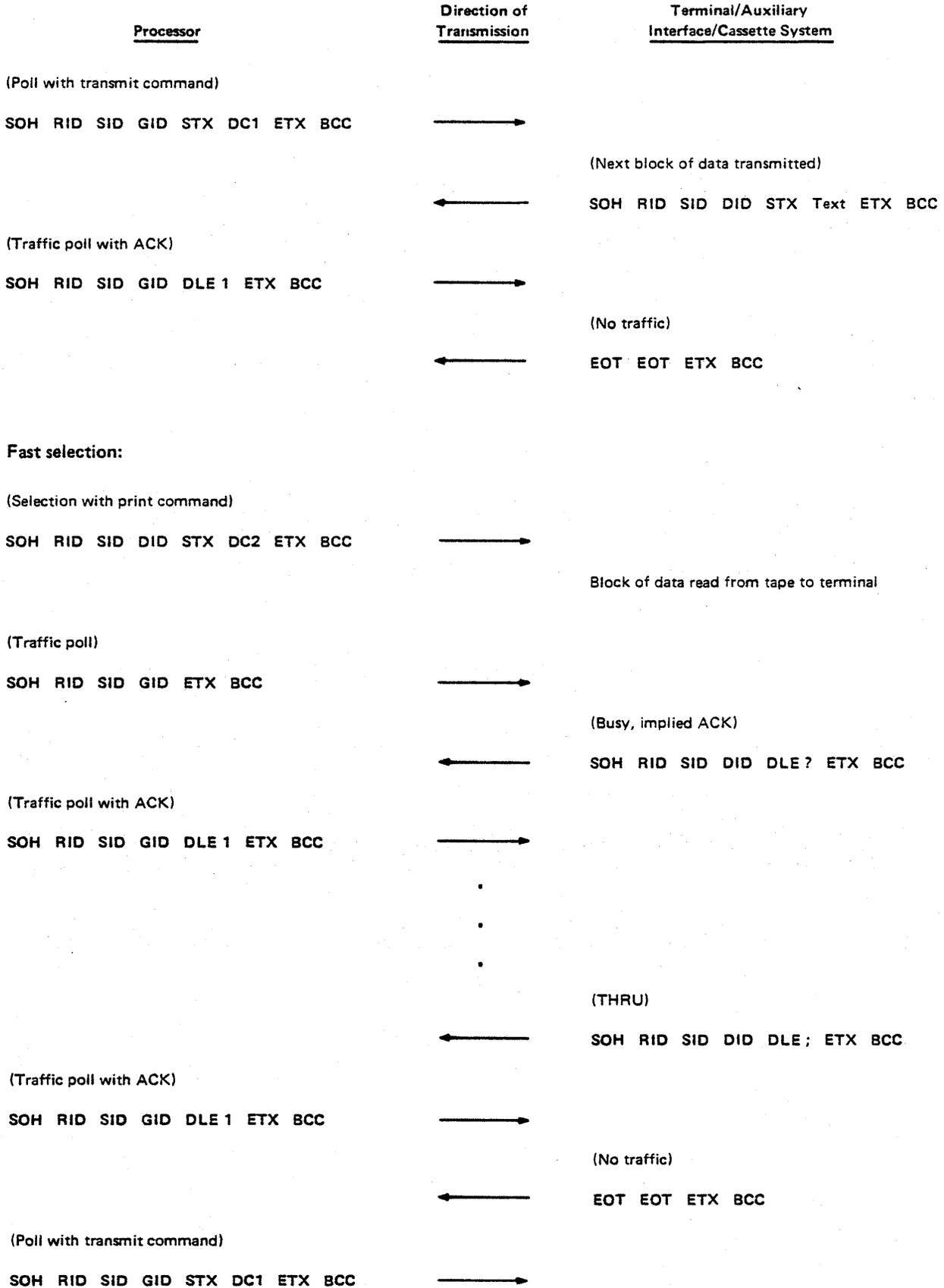
The following examples illustrate typical read sequences, using both the slow-selection and fast-selection methods. It is assumed that reading is to begin at the current tape position. The DID is the read DID for the transport to be used in the operation.

Search, report-address, and backward-one-block operations are handled in the same manner as read operations, except that the processor control commands for these functions are used in conjunction with the print command. (Refer to 4.5.)

#### Slow selection:



<u>Processor</u>	<u>Direction of Transmission</u>	<u>Terminal/Auxiliary Interface/Cassette System</u>
(Traffic poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(No traffic)
	←	EOT EOT ETX BCC
(Poll with transmit command)		
SOH RID SID GID STX DC1 ETX BCC	→	
		(Block of data transmitted)
	←	SOH RID SID DID STX Text ETX BCC
(Traffic poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(No traffic)
	←	EOT EOT ETX BCC
(Message with print command)		
SOH RID SID GID STX DC2 ETX BCC	→	
		Next block of data read from tape to terminal
(Traffic poll)		
SOH RID SID GID ETX BCC	→	
		(Busy, implied ACK)
	←	SOH RID SID DID DLE? ETX BCC
(Traffic poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(THRU)
	←	SOH RID SID DID DLE; ETX BCC
(Traffic poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(No traffic)
	←	EOT EOT ETX BCC



<u>Processor</u>	<u>Direction of Transmission</u>	<u>Terminal/Auxiliary Interface/Cassette System</u>
		(Block of data transmitted)
	←	SOH RID SID DID STX Text ETX BCC
(Poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(No traffic)
	←	EOT EOT ETX BCC
(Next message with print command)		
SOH RID SID GID STX DC2 ETX BCC	→	
		Next block of data read from tape to terminal
(Traffic poll)		
SOH RID SID GID ETX BCC	→	
		(Busy, implied ACK)
	←	SOH RID SID DID DLE? ETX BCC
(Traffic poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
	.	
	.	
	.	
		(THRU)
	←	SOH RID SID DID DLE; ETX BCC
(Traffic poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(No traffic)
	←	EOT EOT ETX BCC
(Traffic poll with transmit command)		
SOH RID SID GID STX DC1 ETX BCC	→	
		(Next block of data transmitted)
	←	SOH RID SID DID STX Text ETX BCC
(Poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(No traffic)
	←	EOT EOT ETX BCC

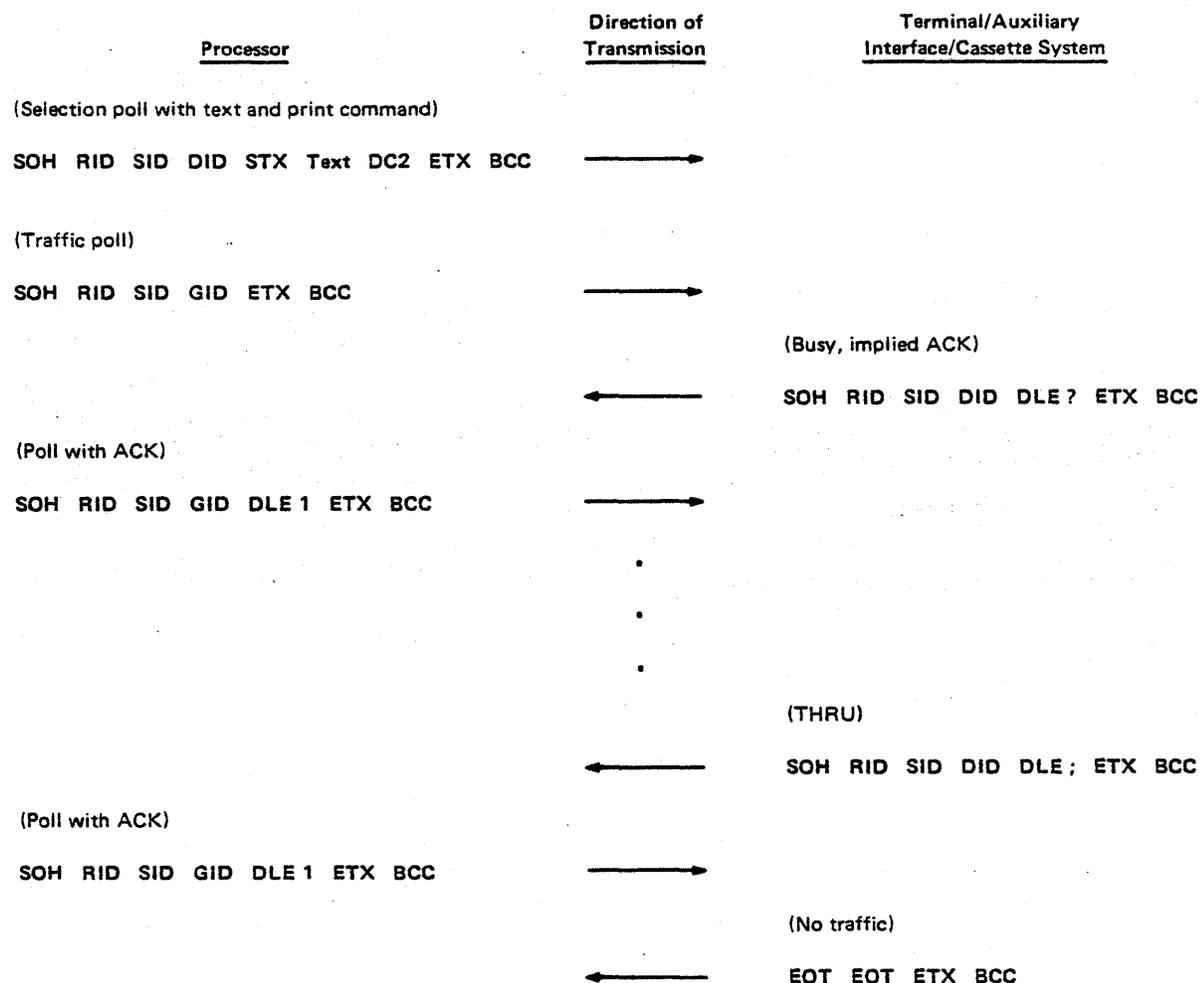
### 4.3.2. Precautions

1. After a rewind, it is recommended that a report-address command (4.5.2) be issued to position the tape at read load point.
2. A cursor wraparound condition, which results when a full screen of data is read back to the UNISCOPE terminal, may cause some inconvenience in transferring data to a processor or another device such as a printer. For an explanation of this condition, and the recommended precaution, refer to 4.2.2. Tape cassette systems used with terminals containing auxiliary interface feature 1247-01 are not affected by this condition.

## 4.4. AUTOMATIC TRANSMIT

The cassette system has an automatic transmit function. When enabled by operator control, this function allows the cassette system to insert the transmit code (DC1 or ESC DC1, depending on how the unit is strapped) at the end of a data block during a read operation. With the automatic transmit function enabled, it is not necessary for the processor to send a transmit code. The contrast between read operations with and without automatic transmit is shown in the following sequences (the DID is the read DID):

Without automatic transmit:



<u>Processor</u>	<u>Direction of Transmission</u>	<u>Terminal/Auxiliary Interface/Cassette System</u>
(Poll with transmit command)		
SOH RID SID GID STX DC1 ETX BCC	→	
(Traffic poll)		
SOH RID SID GID ETX BCC	→	
		(Block transmitted)
	←	SOH RID SID DID STX Text ETX BCC
(Poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(No traffic)
	←	EOT EOT ETX BCC
With automatic transmit:		
(Selection and print command)		
SOH RID SID DID STX Text DC2 ETX BCC	→	
(Traffic poll)		
SOH RID SID GID ETX BCC	→	
		(Busy, implied ACK)
	←	SOH RID SID DID DLE ? ETX BCC
(Poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(THRU)
	←	SOH RID SID DID DLE ; ETX BCC
(Traffic poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(Text with automatic transmit)
	←	SOH RID SID DID STX Text ETX BCC
(Poll with ACK)		
SOH RID SID GID DLE 1 ETX BCC	→	
		(No traffic)
	←	EOT EOT ETX BCC

## 4.5. PROCESSOR CONTROL COMMANDS

Cassette system functions other than read and write are initiated from the processor with specific control commands. These control commands are treated as single messages, following a selection by the read DID. A new selection is then required to make subsequent command interpretations. The control commands are:

Backward one block: ▷ BS ▭

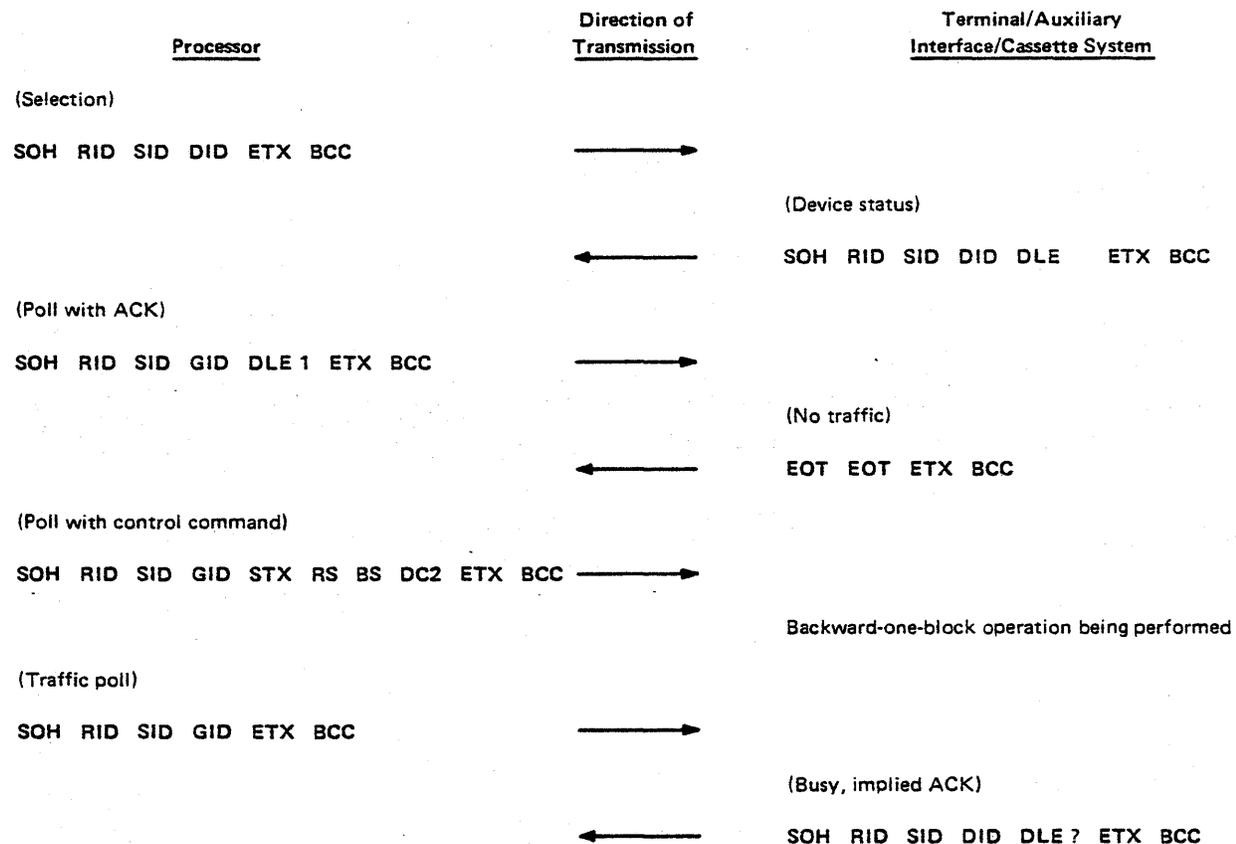
Report address: ▷ VT ▭

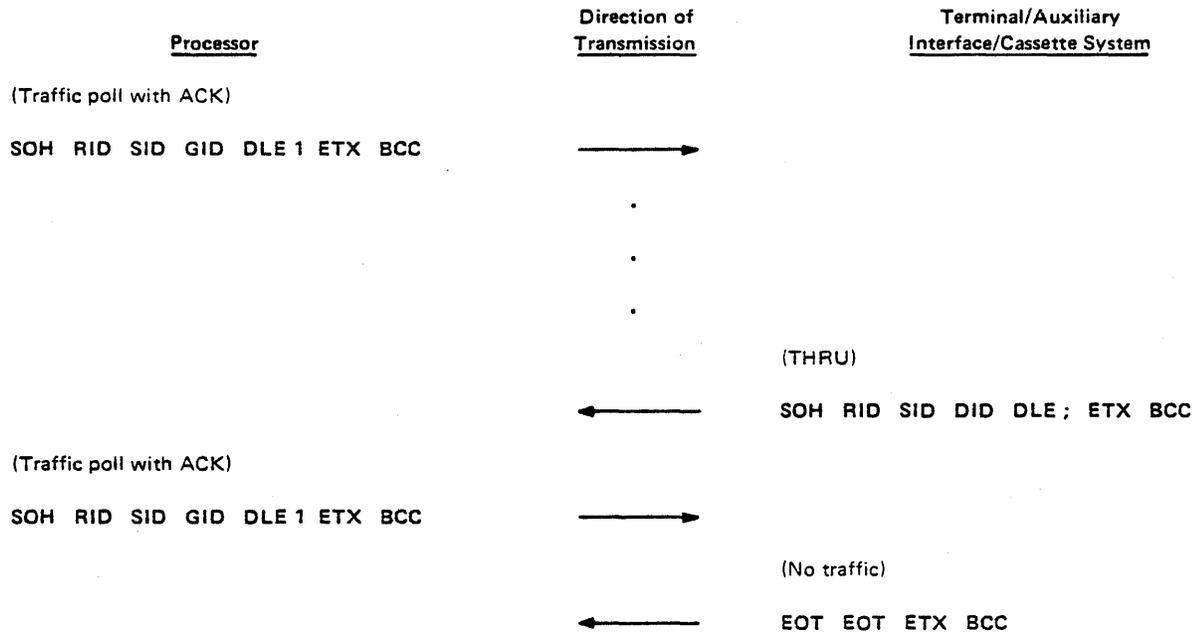
Search (@ and A modes): ▷ CAN mode taaa ▭

Search (B and C modes): ▷ CAN mode t/identifiers ▭

### 4.5.1. Backward One Block

The backward-one-block command, with selection, is accomplished as shown below. In this example, the slow-selection method is used and the DID in the first processor message is the read DID for the applicable transport. If the fast-selection method were to be used, only the last processor message would be required and the GID in that message would be replaced with the specific DID. RS is the SOE code and BS is the backward-one-block command code.



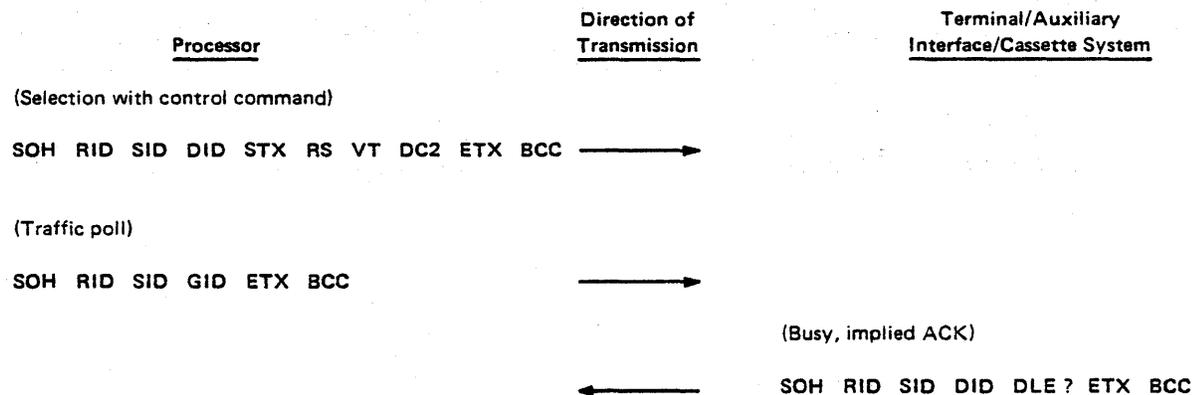


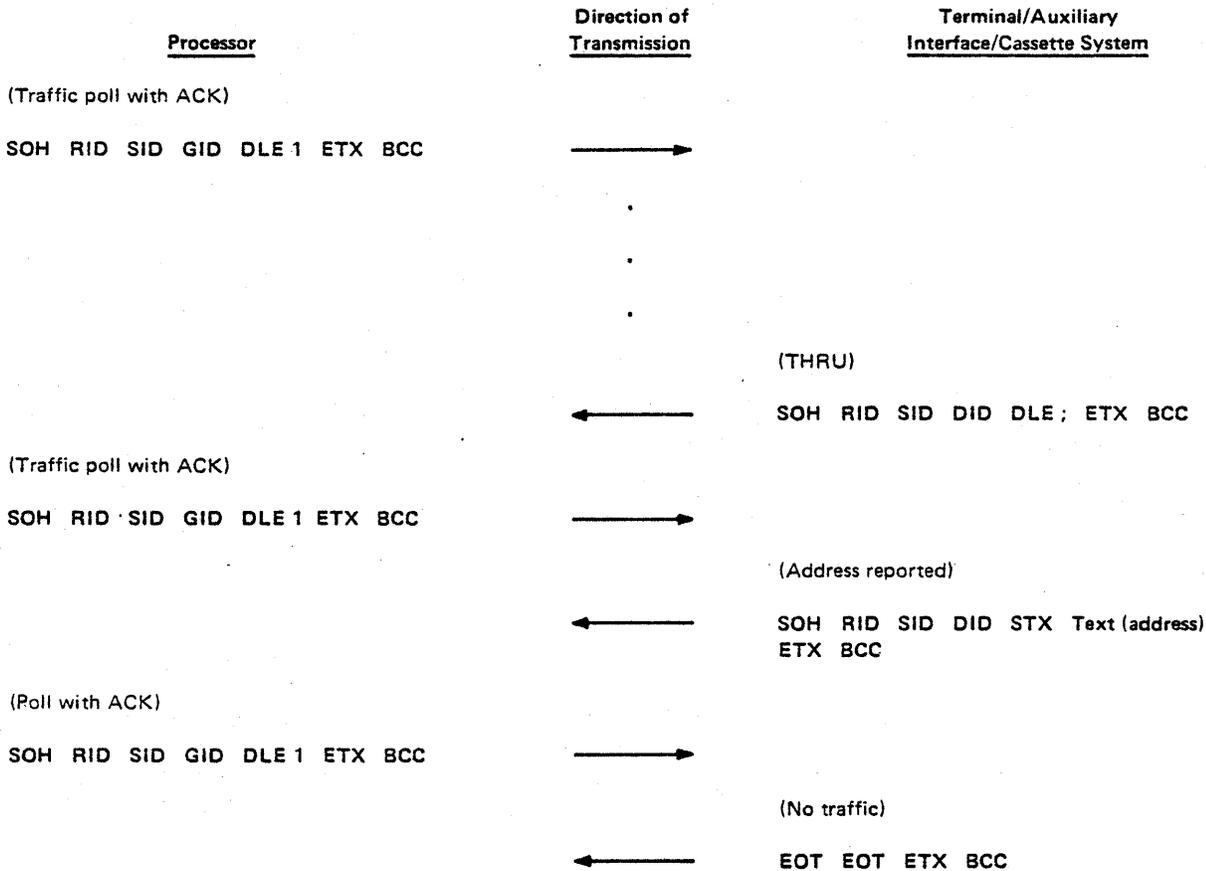
#### 4.5.2. Report Address

The report-address command, with selection, is accomplished as shown below. In this example, the fast-selection method is used and *automatic transmit is assumed*. The processor message is the read DID for the applicable transport. If the slow-selection method were to be used, the processor message could include a GID instead of a DID. RS is the SOE code and VT is the report-address command code.

#### NOTE:

*A report-address command issued to a transport containing a rewind cassette causes the tape to be positioned at read load point, and that point is reported as the address.*

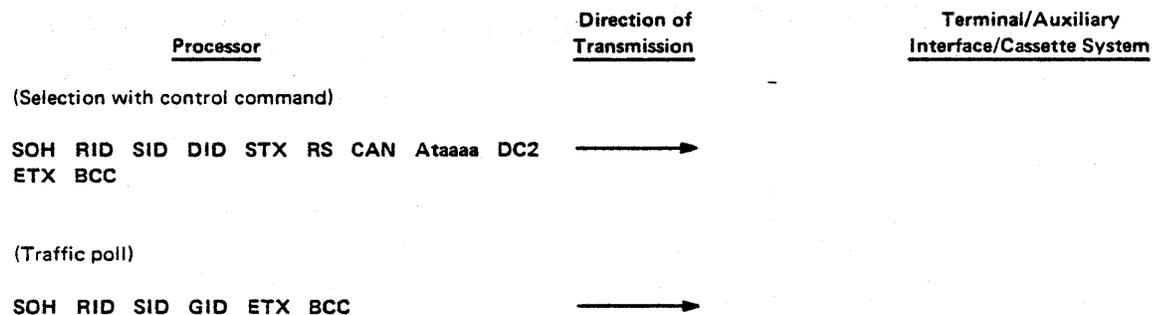


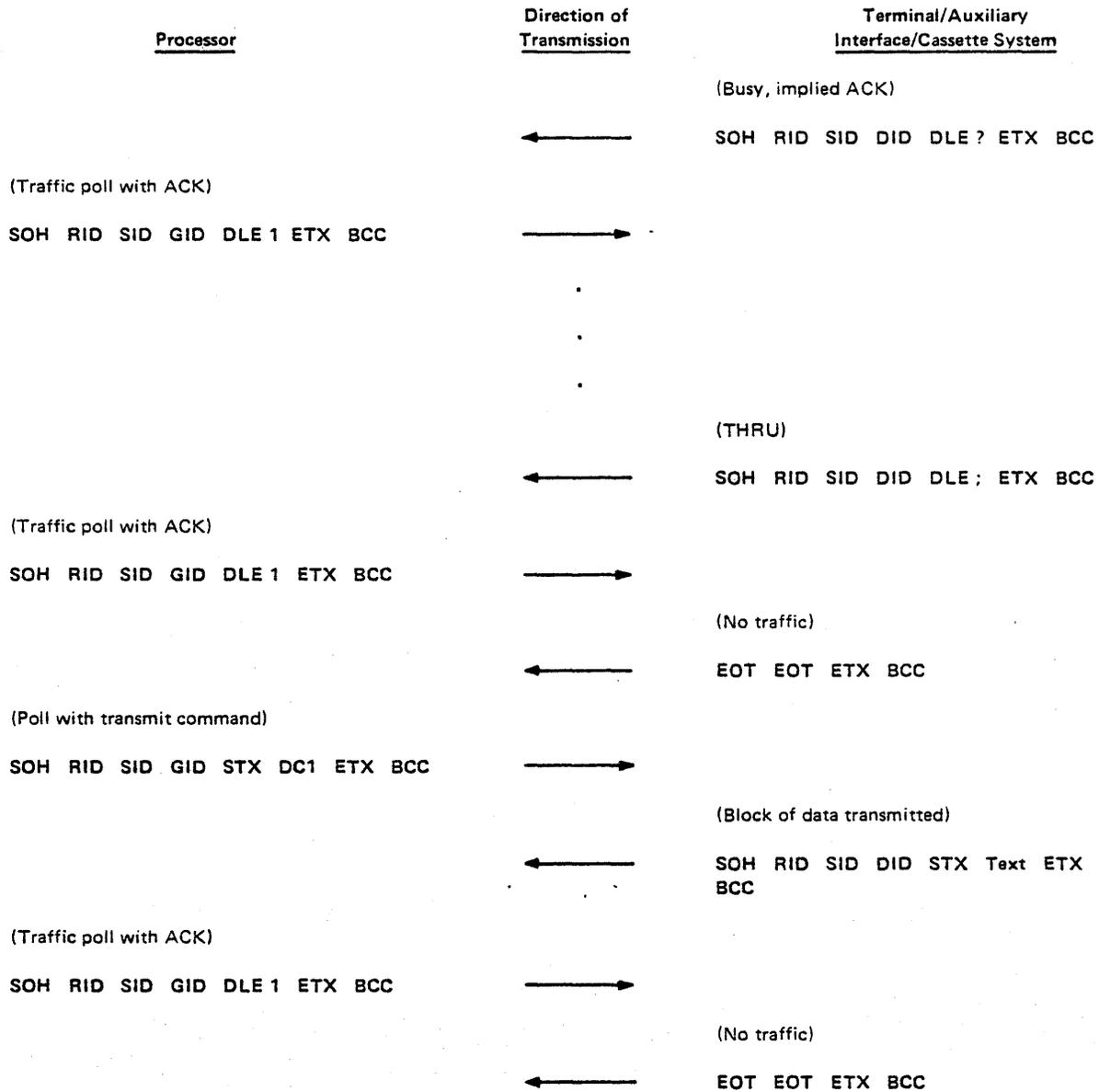


### 4.5.3. Search

The search command follows the same format regardless of the search mode, but the specific information in the command varies, depending on the desired mode.

The search command, with selection, is accomplished as shown below. In this example, the fast-selection method is used to initiate a mode A search. The DID in the processor message is the read DID for the applicable transport. If the slow-selection method were to be used, the processor message could include a GID instead of a DID. RS is the SOE code and CAN is the search command code.





Subsequent message sequences would involve reading or writing successive data blocks.

In search modes A, B, and C, when the specified address or identifier set is located, a normal data read is performed. After completion of the read, the cassette system transport stops in the interblock gap just after the block read. However, in the mode @ search, no automatic data read occurs. The transport stops at the requested address and the cassette system automatically deselects.

## 4.6. USE OF SEARCH COMMANDS

### 4.6.1. Types and Command Formats

The cassette system can search in four different modes:

- Mode @

This search is initiated by the following statement:

```
RS @taaaa DC2
```

RS is the SOE or home position, @ is the mode of search, t is the tape track to be searched (1 or 2), and aaaa is the 4-digit address. The print command (DC2) is made immediately following the address digits without repositioning the cursor.

The cassette system drives the tape at high speed to a position of 20 address counts before the indicated address, then proceeds at read speed to the address, and automatically deselects when the address is reached. This search positions the tape at the count of the supplied address, but orientation to data blocks previously placed on tape is only approximate. To orient to previous blocks, perform a backward-one-block operation following the mode @ search.

**CAUTION**

*After the tape stops in an @-type search, caution should be used in selection of the next operation to ensure that an inadvertent write selection is not made. Selection of the write function will erase any tape data under the read/write head, even before the write operation is performed.*

The cassette system also recognizes 0 or ` in place of @ in the search command statement.

- Mode A

This search is initiated by the following statement:

```
RS Ataaaa DC2
```

RS is the SOE or home position, A is the mode of search, t is the tape track to be searched (1 or 2), and aaaa is the 4-digit address. The print command (DC2) is made immediately following the address digits without repositioning the cursor.

The cassette system drives the tape at high speed to a position of 20 address counts before the address indicated in the command. The search then continues at read speed to compare the command-supplied address with the address in the data block heading. A read of the block is performed as part of the search command when the address is reached. If the address in the search command statement is not found, the cassette system will continue to search at read speed to the end of tape.

The cassette system also recognizes 1 or a in place of A in the search command statement.

## ■ Mode B

This mode can be used only with units equipped with the identifier-search feature. The search is initiated by the following statement:

```
RS Btaaaa/identifiers DC2
```

RS is the SOE or home position, B is the mode of search, t is the tape track to be searched (1 or 2), aaaa is the 4-digit address, / indicates the beginning of the identifiers, and the identifiers are the data characters (up to 16) which identify the desired location. The print command (DC2) is made immediately following the identifiers without repositioning the cursor.

The cassette system drives the tape at high speed to a position of 20 address counts before the address indicated in the command. The search then continues at read speed to compare the identifiers with an equal number of characters at the first of each block until an exact match is found. A read of the identified block is performed as part of the search command. If a match is not found, the search will continue at read speed to the end of tape.

The cassette system also recognizes 2 or b in place of B in the search command statement.

**NOTE:**

*If protected data is included in the identifier-search mask, the mode B search will not work.*

## ■ Mode C

This mode can be used only with units equipped with the identifier-search feature. The search is initiated by the following statement:

```
RS Ct/identifiers DC2
```

RS is the SOE or home position, C is the mode of search, t is the tape track to be searched (1 or 2), / indicates the beginning of the identifiers, and the identifiers are the data characters (up to 16) which identify the desired location. The print command (DC2) is made immediately following the identifiers without repositioning the cursor. Address digits are not required in this search mode. As a convenience for programming, they may be used to avoid providing an additional search command form. If they are used, they may be all zeros or any dummy figures desired; they will be ignored as part of the search statement.

The cassette system drives the tape at read speed to compare the identifiers with an equal number of characters at the first of each block until an exact match is found. A read of the identified block is performed as part of the search command.

The cassette system also recognizes 3 or c in place of C in the search command statement.

**NOTE:**

*If protected data is included in the identifier-search mask, the mode C search will not work.*

#### 4.6.2. Special Considerations

1. An online mode @ search to address 0000 will initiate a rewind.
2. A search to an address between 0001 and 0020 is invalid and will place the auxiliary interface in a sustained busy condition (that is, since no ETX is received from the cassette system, a DLE ? is returned to the processor).
3. When the mode @ search is used on a tape cassette that has had information written over older information, the cassette system may locate a position that contains enough of the older information (such as a preamble or a postamble) to cause a misreading of a valid interblock gap. To avoid this problem, follow the mode @ search with a backward-one-block operation to find a valid interblock gap, noting the address of this block for future mode A searches.

**NOTE:**

*A recommended procedure for reusing tape cassettes when the data is no longer needed is to degauss them, removing the information entirely and providing magnetically clean tapes.*

4. When a mode @ search is used to position a tape, a circuit in the cassette system is conditioned to adjust the length of the interblock gap for a write on the cassette following the search. If, however, the other cassette is used before a write on the search-positioned cassette, the circuit will be reset and may produce an interblock gap of improper length.

#### 4.7. END-OF-TAPE PROCEDURES

**NOTE:**

*The end-of-tape procedures described in this paragraph are applicable to cassette systems operating with either UNISCOPE 100 or UNISCOPE 200 terminals. However, cassette system types 0866-02 and -03, when strapped for operation with UNISCOPE 200 terminals, do not use the hole in the tape to sense the end-of-tape condition in a write operation. Instead, the end-of-tape signal is generated (in a write operation) at a tachometer count of 6000. (See 2.8.10 for details.)*

Two end-of-tape (EOT) conditions may be encountered: the end of the first track and the end of the second track. In either case, when the cassette system detects the EOT condition (by sensing the EOT hole or by reaching a tachometer count of 6000), the following sequence occurs:

1. Normal completion of the write or read operation in process
2. Deselection
3. Retention of the status at completion of the last block

<b>CAUTION</b>
----------------

*After EOT is detected, initiating a device deselection (octal 162) online or pressing the STOP switch offline will clear the EOT status and cause the tape to proceed to clear leader on the next operation. If the tape moves to clear leader, an offline rewind will be required to continue operation.*

If no data errors were encountered during the operation in which EOT was determined, a subsequent selection of this transport will result in an EOT status response (2 or 6, Table 4-1) until the tape is repositioned.

If a data error was encountered during the operation, a subsequent selection of the transport will result in a data-error response (3 or 7, Table 4-1), which is cleared after being reported online.

If the tape happens to stop with the EOT hole directly over the EOT sensor, the cassette system will interpret the EOT hole as clear leader and halt operation. An offline rewind will enable continued operation.

## 4.8. ERROR CONDITIONS AND RECOVERY PROCEDURES

A special error code is standardized to be used in error detection. This code is usually the start-blink-field code (octal 034), but the user may specify a different code. The selected code is strapped in by the Sperry Univac customer engineer at the user site.

### 4.8.1. Read Errors

If a data error (parity or timing error) occurs during a read operation, the special error code is immediately supplied to the auxiliary interface and the error status is retained. The read is discontinued and the tape is positioned at the next interblock gap. The cassette system automatically desejects and the auxiliary interface is placed in a sustained busy condition (that is, since no ETX is received from the cassette system, a DLE ? is returned to the processor). At the next selection, the sustained busy condition is cleared in the auxiliary interface, the cassette system returns a data-error response (7, Table 4-1) to the auxiliary interface, and the UNISCOPE terminal sends a DLE : to the processor. After the data-error response has been returned, the error status in the cassette system is cleared; with the next selection, the normal read operational sequence can be continued.

To check the error location, or to read the data block again, reposition the tape by performing a backward-one-block or search operation (4.5.1 and 4.6, respectively).

If the cassette system repeatedly returns an error response at the same location on the tape, the tape is probably damaged or dirty at that point. If it is possible to examine the suspect cassette, first determine the address of the previous block (and exact identifiers, if desired), and then remove the tape cassette from the system at the point of error and examine the tape for damage or contamination. Remove contamination with a cotton swab. (Never touch the magnetic oxide with your fingers.) If nothing removable is found, or if damage is found, bypass the area by advancing the tape to a location beyond the damaged spot for the next write. The mode @ search may be used to specify the start address.

### 4.8.2. Write Errors

If a timing error occurs during a write operation, the special error code is written on the tape and is immediately followed by the normal block termination sequence. The error status is retained. The cassette system automatically desejects and the auxiliary interface is placed in a sustained busy condition. At the next selection, the sustained busy condition is cleared in the auxiliary interface, the cassette system returns a data-error response (3, Table 4-1) to the auxiliary interface, and the UNISCOPE terminal sends a DLE : to the processor. After the data-error response has been returned, the error status in the cassette system is cleared; with the next selection, the tape can be repositioned

to rewrite the data block or the normal write operation can be continued. To correct the write error, perform a backward-one-block operation (or search to a backward position) and then attempt to rewrite the data block containing the error.

**NOTE:**

*In tape cassette system types 0866-00 and -01, use of the print transparent function with 10 or more blank lines will cause a write timing error. The reason for this timing error is that the print transparent function suppresses all cursor return (CR) codes that the UNISCOPE terminal automatically places at the end of each line, blank or not. Thus, because the 10 lines appear to be completely blank, the time between data inputs is too long for the timing provisions of the types 0866-00 and -01 cassette system, and a timing error is called. In tape cassette system types 0866-02 and -03, the timing provisions have been changed to avoid this condition.*

### 4.8.3. Read-After-Write Errors

Units equipped with the optional read-after-write feature (4.9.1) can check for character parity and block parity on a block of data as it is being written. If a parity error is detected in a data character or the block check character, the write/read operation continues until all characters in the data block have been written on tape; then the cassette system desejects and, since the data transfer from the UNISCOPE terminal is complete, the terminal sends a DLE ; (THRU) to the processor. On the next selection, the data-error condition (DLE :) is reported.

If no errors are encountered during the write/read operation, the cassette system will be in the ready state, awaiting another command.

In using the read-after-write feature, programmers must remember that the DLE ; (THRU) response does not guarantee that the data block was written correctly. Therefore, to ensure that each block is written correctly before the data is lost from the terminal memory, it is recommended that a slow selection be performed after each write operation.

### 4.8.4. Improper-Selection Errors

If a new selection of the cassette system (or other device connected to the auxiliary interface) is attempted while a block is being written or read, the system automatically desejects and error status is retained. When the interruption occurs during a write operation, the character being written is completed; then the special error code is written, followed by a normal block termination sequence. When the interruption occurs during a read operation, data transfer stops, the tape is repositioned at the following interblock gap, and then the transport stops.

When the interrupting selection is received, the cassette system returns a device-unable-to-proceed/no-response status code (4 or 8, Table 4-1), and the UNISCOPE 100 terminal sends a DLE = to the processor. This response is returned on selection attempts until the working transport has stopped. After that time, selection of the other transport will result in the return of device-ready status (1 or 5, Table 4-1). A new selection of the halted transport will result in the return of data-error status (3 or 7, Table 4-1). After the data-error status is reported, the error status is cleared; with the next selection, device-ready status will be returned.

**NOTE:**

*The backward-one-block function, and a read operation when there is no data, cannot be interrupted by processor control except by issuing a selection with an octal 162 DID. This process, however, will master clear the cassette system and immediately halt operation.*

## 4.9. USE OF OPTIONAL FEATURES

The optional read-after-write, protected-format, and record-separator-write features are activated with switches on the back of the cassette system.

### 4.9.1. Read After Write

The read-after-write feature provides cassette system verification of character parity and block parity on a block of data as the data is being written. Data is written as the tape passes over the write head; the tape then passes over the read head and is read. If a parity error is detected in a data character or the block check character, the error is reported as explained in the read-after-write error recovery procedures (4.8.3).

### 4.9.2. Protected Format

The protected format feature allows data entered from a UNISCOPE terminal to be recorded on tape as protected data.

In tape cassette systems operating with UNISCOPE 200 terminals or later-model UNISCOPE 100 terminals containing auxiliary interface feature 1247-01, the protected format feature is enhanced as explained in 4.9.2.2. Translation characters, explained in 4.9.2.1, are required for offline operation with all UNISCOPE terminals and may be used if desired for all online operation.

#### 4.9.2.1. Protected Format Translation Characters

Two characters, specified by the user and strapped in by the Sperry Univac customer engineer, are decoded and translated by the cassette system to SO and SI for writing on the tape. The first character, representing SO, must be placed on the UNISCOPE screen immediately before the data to be protected; the second character, representing SI, must be placed on the screen immediately after the data to be protected. Any number of protected fields may be written in a data block.

When the data enclosed by the SO and SI characters is read from tape back to the terminal, that data is protected (assuming the terminal has protected format capability). The text is shifted one position to the left for each character that was translated as SO or SI. The data remains protected in the terminal storage.

**NOTE:**

*The protected format feature does not serve the same function as a write-protected tape; it is possible to write over information enclosed by SO and SI characters on tape.*

#### 4.9.2.2. Protected Format Enhancements Provided by Auxiliary Interface 1247-01

Auxiliary interface feature 1247-01 is always used with UNISCOPE 200 terminals interfaced with cassette systems and may be used with later-model UNISCOPE 100 terminals. When operating online by way of this interface, the cassette system does not require translation characters to bracket protected data because the auxiliary interface is capable of transferring the actual SO and SI characters to the cassette system. Thus, processor-generated SO and SI characters may be used as protected format limiters. When the protected data is sent to the cassette system, actual SO and SI characters are transferred through the auxiliary interface and placed on tape. Since SO and SI characters do not occupy a space in the terminal memory, data read back to the terminal from the tape cassette does not result in a loss of space and shift of text to compensate for space used by translation characters.

Offline, translation characters must still be used as format limiters as explained in 4.9.2.1. (SO and SI cannot be entered in the terminal from the keyboard.) However, when data bracketed by SO and SI is read back from the tape to the UNISCOPE terminal, that data (assuming the terminal has protected format capability) will not only be protected on the terminal, but may be recorded again on tape as protected data bracketed by the actual SO and SI characters.

#### 4.9.2.3. HT (Tab Stop) Control Character Feature

In tape cassette system types 0866-02 and -03, the HT (tab stop) control character can be strapped as a translation character for the SO or SI function. A tab stop location can then be set by the operator, or through processor control, as explained in 3.5.4. Pressing the tab key moves the cursor to the tab stop location and defines the SO or SI location. When data is transferred to the tape, the HT character is decoded and written on tape as an SO or SI character. The tab stop function is lost in the translation.

**NOTE:**

*If the cassette system is operating with the protected format switch on and is strapped for the HT (tab stop) control character feature, the HT character cannot be used as a tab function alone. HT will always define an SO or SI location. Of course, if the device is not operating in the protected format mode, the HT character has no special protected format function.*

#### 4.9.3. Record Separator Writing

The record-separator-writing feature allows the record separators FS, GS, RS, and US to be written within a data block. Any number of the four record separators may be used in a block.

The user specifies four characters to be decoded and translated by the cassette system as FS, GS, RS, and US. These characters are strapped in at the user site by the Sperry Univac customer engineer. The character corresponding to the desired record separator must be placed within the data at a position corresponding to the desired record separator position. One separator is written for each character decoded; multiple separators are written by using multiple characters.

#### 4.9.4. Identifier Search

This feature extends the search capability so that up to 16 characters may be specified for comparison with the first equal number of characters in a data block for block identification. (Refer to 4.6.1.) No specific programming considerations are necessary to use this feature.

#### 4.9.5. Optional Features Used Only in Offline Operation

##### 4.9.5.1. List and Edit

The list and edit features are strictly offline functions; their uses and the operational procedures required are described in the *UNISCOPE Display Terminal Operator's Guide, UP-8147* (current version). No programming considerations are necessary.

#### 4.9.5.2. Print Transparent

This feature enhances the list feature by allowing the printer to print tape data in line lengths determined by the printer format, rather than the format recorded on tape. When the optional circuitry is provided for this feature, the print transparent function can be selected by a switch on the back of the cassette system. With the switch activated and the cassette system in list mode, the cursor return (CR) characters generated by the UNISCOPE terminal at the end of each line of data are suppressed, or become transparent, to the printer.

#### 4.9.5.3. Copy to Address

The copy-to-address feature, which is used only in offline operation, provides the capability to terminate a copy mode at a specific preamble address.

## Appendix A. UNISCOPE Terminal Responses to Polls

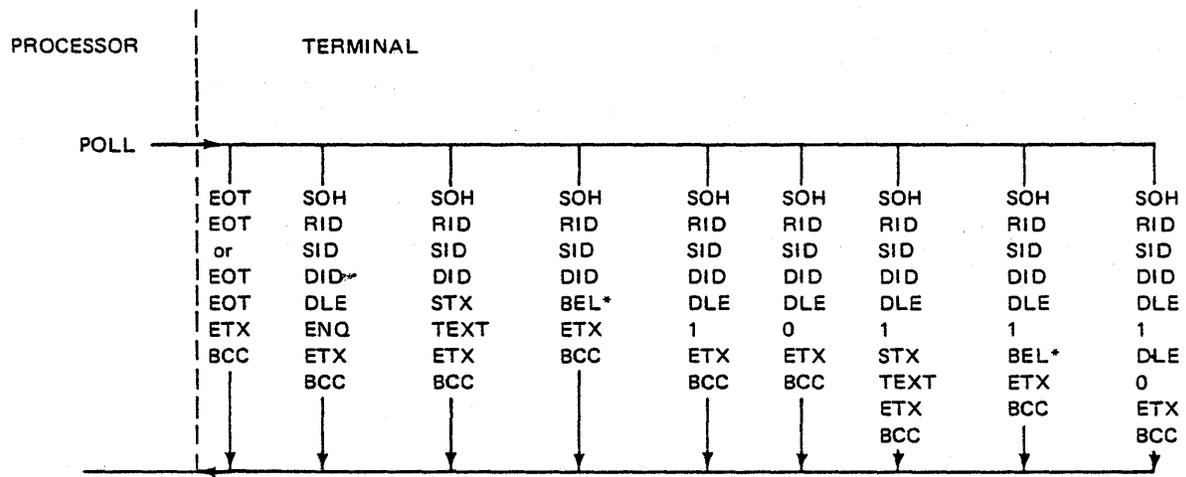
The following illustrations list meaningful UNISCOPE Display Terminal responses to processor polls. These responses are grouped according to the terminal feature and network configurations. The particular response is dependent on the type of poll as well as the state of the terminal. In synchronous systems, all responses will be prefaced by four SYN characters.

Figure A-1 pertains to a UNISCOPE terminal, either single station or on a multiplexer, without an auxiliary interface on the single station or anywhere on the multiplexer.

Figure A-2 pertains to a UNISCOPE terminal that is connected to a multiplexer; the terminal does not have an auxiliary interface but at least one other UNISCOPE terminal on the multiplexer does have an auxiliary interface.

Figure A-3 pertains to a UNISCOPE terminal that has an auxiliary interface feature but is not connected to a multiplexer.

Figure A-4 pertains to a UNISCOPE terminal that has an auxiliary interface feature and is connected to a multiplexer.



\*Can be replaced by the special function key codes 7, G, W, or g.

Figure A-1. Responses by Terminal, Either Single Station or on a Multiplexer, but Without an Auxiliary Interface on the Single Station or Anywhere on the Multiplexer

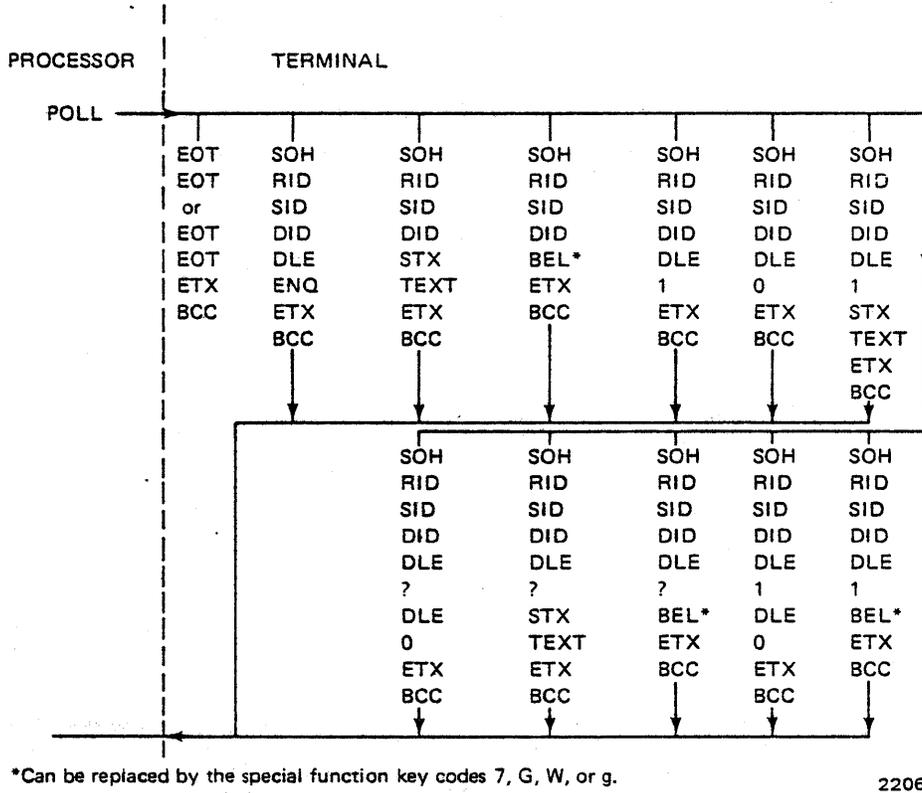


Figure A-2. Responses by Terminal on a Multiplexer With no Auxiliary Interface on the Responding Terminal but With an Auxiliary Interface on Some Other Terminal on the Same Multiplexer

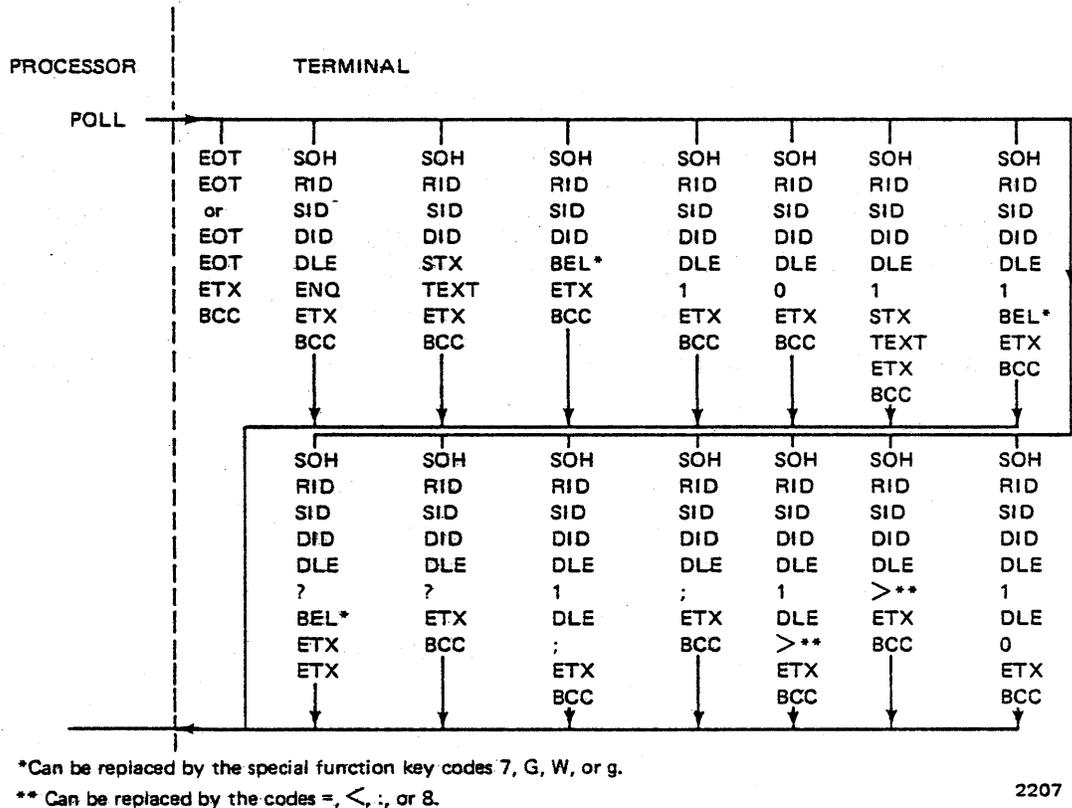
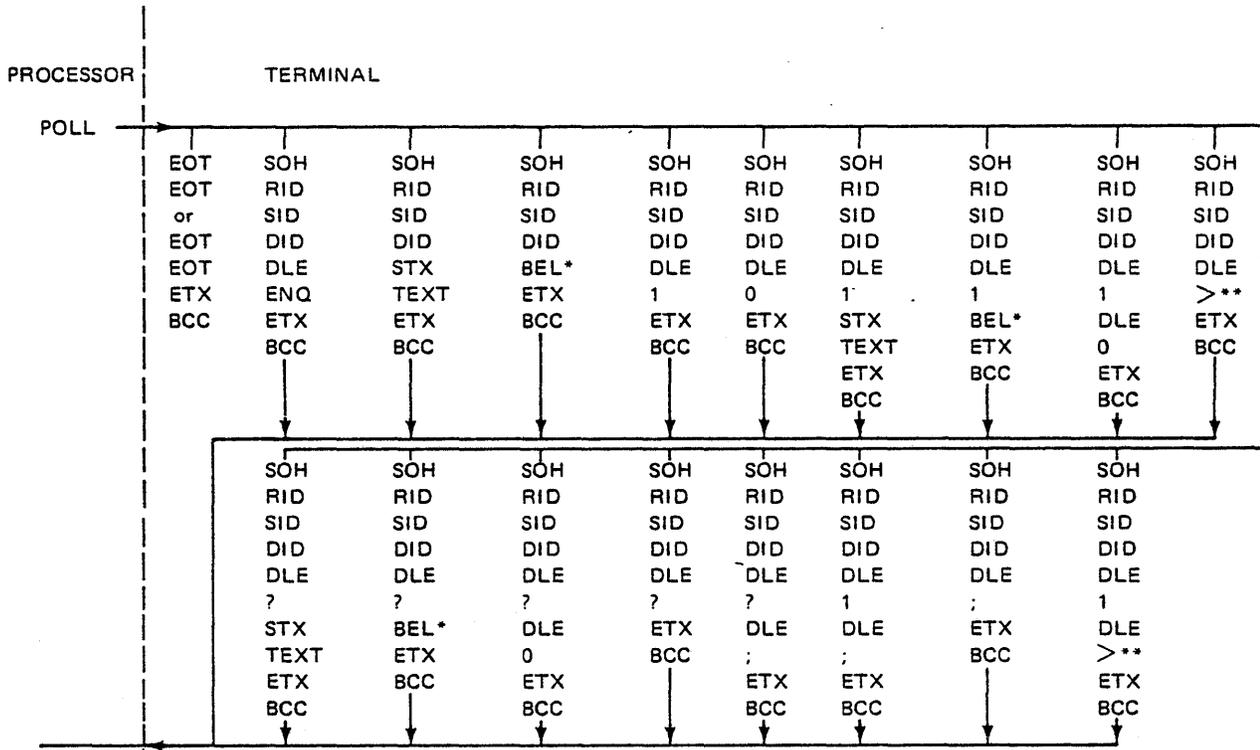


Figure A-3. Responses by Terminal With an Auxiliary Interface but not on a Multiplexer



\*Can be replaced by the special function key codes 7, G, W, or g.

\*\*Can be replaced by the codes =, <, :, or 8.

Figure A-4. Responses by Terminal With an Auxiliary Interface and on a Multiplexer

## Appendix B. Line Control Rules

### B.1. GENERAL

Certain rules must be followed to allow recovery from communications line error without loss or duplication of messages. These rules establish the sequence in which the various transmissions are exchanged. Some of these rules reflect hardware functions and others reflect software functions. Figure B-1 illustrates the traffic flow between the UNISCOPE Display Terminal and the processor.

### B.2. HARDWARE RULES

1. The terminal responds only to error-free polls. (The only exceptions are DNAK and the mandatory disconnect message, DLE EOT.)
2. If a terminal that is owed an acknowledgment does not receive an acknowledgment within the next good poll that it recognizes, the terminal sends a reply-request (DLE ENQ) message.
3. The terminal expects an acknowledgment to any message containing an SOH (any message other than the no-traffic-without-acknowledge, EOT EOT). DNAK may be sent in response to the reply-request message.
4. The terminal acknowledges any message containing an STX (processor text message). If the terminal is on a multiplexer, this acknowledgment may be passed to another terminal.
5. The terminal never tries to send two successive text messages. Thus, a terminal does not respond with a text message to a poll that includes the acknowledgment to its previous text message. When the terminal is on a multiplexer, it is possible for the processor to receive two text messages from a terminal without an intervening message from that terminal because traffic from another terminal on that multiplexer has taken precedence over the no-traffic condition.
6. A message that causes a busy condition at a terminal is acknowledged with a WABT (DLE ?) sequence in response to the poll that follows the message. The response to subsequent specific polls is DLE ? as long as the busy condition persists; the no-traffic message is sent in response to subsequent general polls. The end of a busy condition is indicated by transmission of the THRU (DLE ;) sequence from the terminal. The DLE ; response does not contain text but may contain an acknowledgment or a busy message from another terminal on the multiplexer. However, slow polling rates and/or fast data transfers can create situations where DLE 1 plus DLE ; could be returned without DLE ? first being returned to the processor.
7. If the terminal has more than one message to send, the following priority determines which will be sent:
  - a. Device status
  - b. Text
  - c. Function key or BEL message

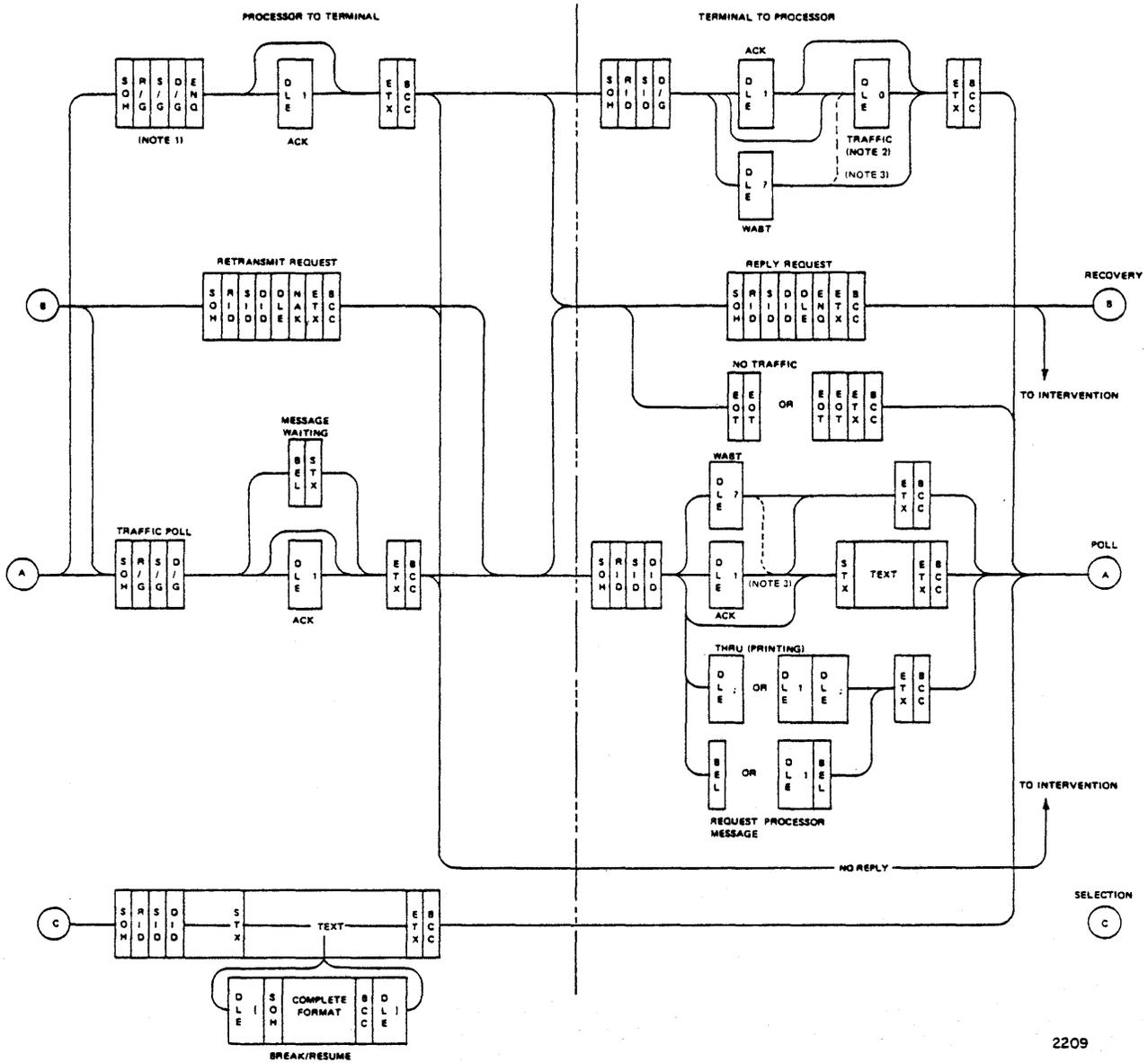
### B.3. SOFTWARE RULES

1. Acknowledgments from the processor to the terminal are always sent with the poll. The normal response to an acknowledgeable message from the terminal is a poll-with-acknowledge message.
2. After an acknowledgeable message is received from a terminal, a poll-with-acknowledge message must be sent to that terminal before a text message can be sent to that terminal.
3. The processor treats any error in a message as a no-response condition, and because the processor expects a reply to every poll, it will repeat the poll that preceded the no-response condition. (Any acknowledgment included with the original poll will be eliminated and a general DID will be used when the poll is repeated.)
4. The processor response to a reply request (DLE ENQ) from a terminal is:
  - a. A poll-with-acknowledge message if the last message correctly received from the terminal sending the reply request was a text message. That text message should have been previously acknowledged.
  - b. A retransmission request (DLE NAK) if the last message correctly received from the terminal sending the reply request was an acknowledgeable message other than text. That acknowledgeable message should have been previously acknowledged.
  - c. A retransmission request (DLE NAK) if the terminal sending the reply request is other than one to which an acknowledgment had just been sent.
5. The retransmission request (DLE NAK) is sent only in response to a reply request (DLE ENQ) from a terminal and is specifically addressed to that terminal.
6. The processor uses specific addresses (specific RID, specific SID) during error recovery.
7. A sustained busy during an auxiliary interface transfer should be followed by a device selection to obtain device status and determine the cause of the sustained busy.

Although Figure B-1 is a useful guide in planning for a system using the UNISCOPE Display Terminal, its use requires a thorough understanding of the codes and formats discussed in Section 3, particularly 3.3 and 3.4. Points to remember in using the figure are:

1. The processor is represented on the left side.
2. The terminal is represented on the right side.
3. All traffic flow is from left to right; the A, B, and C on the right return the flow to the corresponding letters on the left for continuation as appropriate.
4. All possible responses to any given sequence are shown; however, any given system may not be programmed to use all these responses.

In using the figure, refer to 3.3 and 3.4 until an understanding is developed of how the traffic flow takes place, as represented in Figure B-1.



2209

NOTES.

1. R/G signifies RID or GID; S/G signifies SID or GID; D/G signifies DID or GID.
2. DLE 0 is replaced by DLE >, DLE <, DLE !, DLE \*, or DLE 8 when device status is pending.
3. In multiplexer operation.

Figure B-1. UNISCOPE Terminal Traffic Flow Diagram

## Appendix C. Operation Sequence Examples

### C.1. GENERAL

The examples that follow indicate the required sequence of operation. Transmissions from the processor are on the left of the page with an arrow to the right (→). Transmissions from the terminal or terminals are on the right of the page with an arrow to the left (←). To indicate the occurrence of a transmission error (line hit), the appropriate arrows are marked with an X (→X or ←X).

Characters used in the examples are described in 3.3 and 3.4. The text format and code definitions, which are also applicable here, are described in 3.5 and 3.6.

### C.2. SINGLE STATION (PERFECT TRANSMISSION)

In the following sequences, the station address is: RID = 1, SID = a, DID = p.

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
1.			Operator enters information and requests information. Operator presses TRANSMIT.
2.	(Specific status poll) SOH 1 a p ENQ ETX BCC	→	
3.		←	(Traffic indication) SOH 1 a p DLE 0 ETX BCC
4.	(Specific traffic poll with ACK) SOH 1 a p DLE 1 ETX BCC	→	
5.		←	(Text) SOH 1 a p STX TEXT ETX BCC
6.	(Specific traffic poll with ACK) SOH 1 a p DLE 1 ETX BCC	→	

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
7.		←	(No traffic) EOT EOT
8.	(Text) SOH 1 a p STX TEXT ETX BCC	→	
9.	(Specific traffic poll) SOH 1 a p ETX BCC	→	
10.		←	(No traffic with ACK) SOH 1 a p DLE 1 ETX BCC
11.	(Specific traffic poll with ACK) SOH 1 a p DLE 1 ETX BCC	→	
12.		←	(No traffic) EOT EOT

### C.3. SINGLE STATION WITH AUXILIARY INTERFACE DEVICE (PERFECT TRANSMISSION)

In the following sequences, the station address is: RID = 1, SID = a, DID = s (printer).

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
1.			Operator enters information and requests information to be printed out. Operator presses TRANSMIT.
2.	(Specific traffic poll) SOH 1 a p ETX BCC	→	
3.		←	(Text) SOH 1 a p STX TEXT ETX BCC
4.	(Specific traffic poll with ACK) SOH 1 a p DLE 1 ETX BCC	→	
5.		←	(No traffic) EOT EOT

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
6.	(Specific traffic poll) SOH 1 a s ETX BCC (Specific DID for device selection)	→	
7.		←	(Device status - ready) SOH 1 a s DLE > ETX BCC
8.	(Specific traffic poll with ACK) SOH 1 a p DLE 1 ETX BCC	→	
9.		←	(No traffic) EOT EOT
10.	(Text with PRINT command) SOH 1 a p STX TEXT + DC2 ETX BCC	→	
11.			Terminal receives message correctly and starts printing.
12.	(Specific traffic poll) SOH 1 a p ETX BCC	→	
13.		←	(Auxiliary interface busy - WABT; implied ACK) SOH 1 a s DLE ? ETX BCC
14.	(Specific traffic poll with ACK) SOH 1 a p DLE 1 ETX BCC	→	
15.		←	WABT (still busy) SOH 1 a s DLE ? ETX BCC
16.	(Specific traffic poll with ACK) SOH 1 a p DLE 1 ETX BCC	→	
17.		←	THRU (auxiliary interface transfer complete) SOH 1 a s DLE ; ETX BCC

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
18.	(Specific traffic poll with ACK) SOH 1 a p DLE 1 ETX BCC	→	
19.		←	(No traffic) EOT EOT

#### C.4. MULTISTATION WITH TWO TERMINALS ON A MULTIPLEXER (PERFECT TRANSMISSION)

In the following sequences, the station addresses are: For terminal a, RID = 1, SID = a, DID = p; for terminal b, RID = 1, SID = b, DID = p.

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
1.			Operator of terminal a presses TRANSMIT
2.	(General status poll) SOH 1 P p ENQ ETX BCC	→	
3.		←	(Traffic indication from terminal a) SOH 1 a p DLE 0 ETX BCC
4.			Operator of terminal b presses TRANSMIT.
5.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
6.		←	(Text from terminal a) SOH 1 a p STX TEXT ETX BCC
7.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
8.		←	(Text from terminal b) SOH 1 b p STX TEXT ETX BCC
9.	(Text to terminal a) SOH 1 a p STX TEXT ETX BCC	→	

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
10.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
11.		←	(No traffic with ACK from terminal a) SOH 1 a p DLE 1 ETX BCC
12.	(Text to terminal b) SOH 1 b p STX TEXT ETX BCC	→	
13.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
14.		←	(No traffic with ACK from terminal b) SOH 1 b p DLE 1 ETX BCC
15.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
16.		←	(No traffic) EOT EOT

**C.5. MULTISTATION WITH ACKNOWLEDGMENT PASSED BY MULTIPLEXER (PERFECT TRANSMISSION)**

In the following sequences, the configuration and addresses are the same as in C.4.

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
1.			Operator of terminal a presses TRANSMIT
2.	(General traffic poll) SOH 1 P p ETX BCC	→	
3.		←	(Text from terminal a) SOH 1 a p STX TEXT ETX BCC
4.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
5.		←	(No traffic) EOT EOT
6.	(Text to terminal a) SOH 1 a p STX TEXT ETX BCC	→	
7.		←	Operator of terminal b presses TRANSMIT
8.	(General traffic poll) SOH 1 P p ETX BCC	→	
9.		←	(Text from terminal b with ACK from terminal a) SOH 1 b p DLE 1 STX TEXT ETX BCC
10.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
11.		←	(No traffic) EOT EOT
12.	(Text to terminal b) SOH 1 b p STX TEXT ETX BCC	→	
13.	(General traffic poll) SOH 1 P p ETX BCC	→	
14.		←	(No traffic with ACK from terminal b) SOH 1 b p DLE 1 ETX BCC
15.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
16.		←	(No traffic) EOT EOT

### C.6. MULTISTATION – TWO TERMINALS WITH AUXILIARY INTERFACE DEVICES ON A MULTIPLEXER (PERFECT TRANSMISSION)

In the following sequences, the station addresses are: For terminal a, RID = 1, SID = a, DID = s (printer); for terminal b, RID = 1, SID = b, DID = s (printer).

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
1.			Operator of terminal a presses TRANSMIT
2.	(General traffic poll) SOH 1 P p ETX BCC	→	
3.		←	(Text from terminal a) SOH 1 a p STX TEXT ETX BCC
4.	(Specific traffic poll with ACK to terminal a; specific DID for device selection) SOH 1 a s DLE 1 ETX BCC	→	
5.		←	(Device status – ready) SOH 1 a s DLE > ETX BCC
6.			Operator of terminal b presses TRANSMIT
7.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
8.		←	(Text from terminal b) SOH 1 b p STX TEXT ETX BCC
9.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
10.		←	(No traffic) EOT EOT
11.	(Text with print code to terminal a) SOH 1 a p STX TEXT + DC2 ETX BCC	→	

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
12.	(General traffic poll SOH 1 P p ETX BCC	→	
13.		←	(WABT from terminal a) SOH 1 a s DLE ? ETX BCC
14.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
15.		←	(No traffic) EOT EOT
16.	(Text with print code to terminal b; specific DID for device selection) SOH 1 b s STX TEXT + DC2 ETX BCC	→	
17.			Terminal a finishes printing
18.	(General traffic poll) SOH 1 P p ETX BCC	→	
19.		←	(THRU from terminal a with WABT from terminal b) SOH 1 a s DLE ? DLE ; ETX BCC
20.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
21.		←	(No traffic) EOT EOT
22.			Terminal b finishes printing
23.	(General traffic poll) SOH 1 P p ETX BCC	→	
24.		←	(THRU from terminal b) SOH 1 b s DLE ; ETX BCC

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
25.	(General traffic poll with ACK) SOH 1 P p DLE 1 ETX BCC	→	
26.		←	(No traffic) EOT EOT

## Appendix D. Error Recovery

### D.1. GENERAL

The UNISCOPE Display Terminal is designed to allow recovery from communications line errors without loss or duplication of messages. To do this, certain error-recovery procedures must be used. The following discussion explains the reason for the procedures used and illustrates with examples some of the error recovery procedures. Appendix B lists the line control rules that are basic to the UNISCOPE terminal operation. Basic in error recovery is the rule that a terminal will not reply with a text message to a poll that includes the acknowledge to its previous text message (B.2, rule 5).

#### D.1.1. Alternating Versus Single Acknowledges (DLE 0 – DLE 1)

Alternating acknowledges are not used for interactive transfer because the single acknowledge improves efficiency and reduces processor storage needs. With alternating acknowledges, the acknowledge status for the last message sent or received, as well as the address, must be stored for every terminal on the line.

Using single acknowledges and hardware rule 5 (B.2), it is necessary to store three conditions for only the last terminal that sent input. These conditions are:

- Whether the last transmission to that terminal was an acknowledge.
- Whether the last input transmission was text.
- The address of the last terminal that sent input.

That is, the status of just one terminal is required instead of each terminal, as is required in the case of alternating acknowledges.

#### D.1.2. Two-Character Responses

The 2-character format for acknowledge, negative acknowledge, reply request, and end of transmission is intended to reduce the incidence of changed meaning resulting from noise. This format is applicable to batch operation, such as that used with the SPERRY UNIVAC DCT 1000 Data Communications Terminal, where the responses are sent back without the benefit of a block check character. The UNISCOPE terminal uses the same format to be compatible with the DCT 1000 when the units are operating together on an interactive network.

### D.1.3. Negative Acknowledge (DPAK – DLE NAK)

The negative acknowledge is the retransmission request. It is sent only from a processor to terminals and never in the reverse direction when operating in an interactive multipoint mode. This is to avoid negative acknowledges from more than one terminal in the event an address was changed by communications line noise. The terminal will treat any message containing an error as invalid and will not respond. Since batch terminals, such as the DCT 1000, have established correct identity before message transfer, a negative acknowledge is returned by these stations without adverse effect. The negative acknowledge always calls for the retransmission of the previous message. The retransmission request is sent with a specific address to avoid the possibility of losing ACKs or WABTs.

### D.1.4. Reply Request (DENQ – DLE ENQ)

The reply request is used by the terminal to solicit retransmission of a missing response from the processor. It is not sent by the processor to the terminal. Instead, a processor, failing to get a response to a poll, polls the same location again without repeating any previous acknowledgments. If an acknowledge is needed, the terminal will send the reply request. Then, whatever caused the no-response, that is, an error to the processor or an error from the processor, recovery will not duplicate or lose a message. A duplicate message results if the error is to an acknowledge from the processor, if a negative acknowledge is the first sequence sent after timeout.

### D.1.5. Timers (Processor)

The timer is used to initiate error recovery. This is necessary since the terminal will not respond to any errored message. It also covers the case where the processor could not recognize a response. The timeout discussed here is the response timer. Where a dialed line is used, a second timer is necessary to guard against hang-ups of the remote equipment. The time period of this second timer must be longer than the response timer. The response timer is started after transmitting ETX BCC for any message for which a reply is expected. The timer is stopped upon receipt of a valid reply from the remote terminal.

### D.1.6. After the Timeout

Hardware rule 5 (B.2) provides for an orderly arrangement of responses so that when a timeout occurs, recovery is accomplished in an orderly fashion. However, this does not apply to status polls. For the status poll to be effective, the next traffic poll following must be able to obtain traffic rather than the forced EOT. As a result, in some error cases, a status poll response is repeated; but this is not as important as a duplicated message. The tradeoff here is an occasional extra exchange with a duplicate status response under error conditions versus an extra exchange after every status poll under normal conditions.

## D.2. EXAMPLES OF ERROR RECOVERY

The following examples illustrate proper error recovery procedures for a variety of conditions. In the examples and accompanying text, the following terms are used for brevity:

GP	—	General traffic poll
ACK	—	Acknowledge (DLE 1)
DENQ	—	Reply request
DNAK	—	Retransmission request
EOT	—	No-traffic response
SP	—	Specific traffic poll
TEXT 1 a	—	Message to or from a terminal. Number is given when more than one message is involved in a series of transmissions. The terminal address is given when more than one terminal is involved in the example.
—X—	—	Hit; that is, parity error caused by noise on the line or other transient cause for erroneous reception of a transmission.

### D.2.1. Hardware Rule 5 Not Followed

Refer to B.2 for hardware rule 5.

<u>Sequence</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
1.	(GP) SOH 1 P p ETX BCC	→	
2.		←	SOH 1 a p STX TEXT 1 ETX BCC
3.	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	→	
4.		←	SOH 1 a p STX TEXT 2 ETX BCC
5.	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	→	
6.		←X—	SOH 1 a p STX TEXT 3 ETX BCC
7.	Timeout		
8.	(GP) SOH 1 P p ETX BCC	→	
9.		←	(DENQ) SOH 1 a p DLE ENQ ETX BCC
10.	What should the processor send now? ACK with GP? DNAK?		

If the processor sends an ACK with GP, the terminal will treat this response as pertaining to TEXT 3, and TEXT 4 will be sent (TEXT 3 will be lost). If the processor sends DNAK, TEXT 3 will be repeated. If, however, the ACK with GP (sequence 5) had been hit instead of TEXT 3, a timeout would also have been caused, and the DNAK would result in retransmission of TEXT 2, causing message duplication. In this example, the processor cannot determine the difference between the loss of an ACK or the loss of text.

**D.2.2. Hardware Rule 5 Correctly Used With One Terminal Active on Multiplexer**

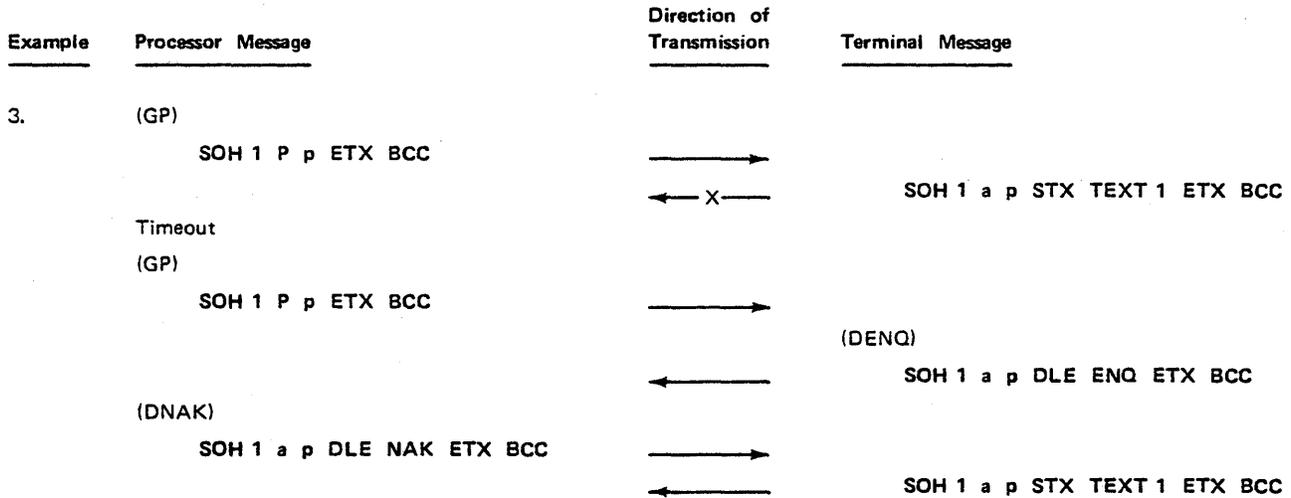
In the following examples, one terminal is active on a multiplexer and all other terminals on the multiplexer are temporarily inactive. (Refer to B.2 for hardware rule 5).

<u>Example</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
1.	(GP) SOH 1 P p ETX BCC	→	
	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	←	SOH 1 a p STX TEXT 1 ETX BCC
	(GP) SOH 1 P p ETX BCC	→	EOT EOT
	Etc.	←	SOH 1 a p STX TEXT 2 ETX BCC

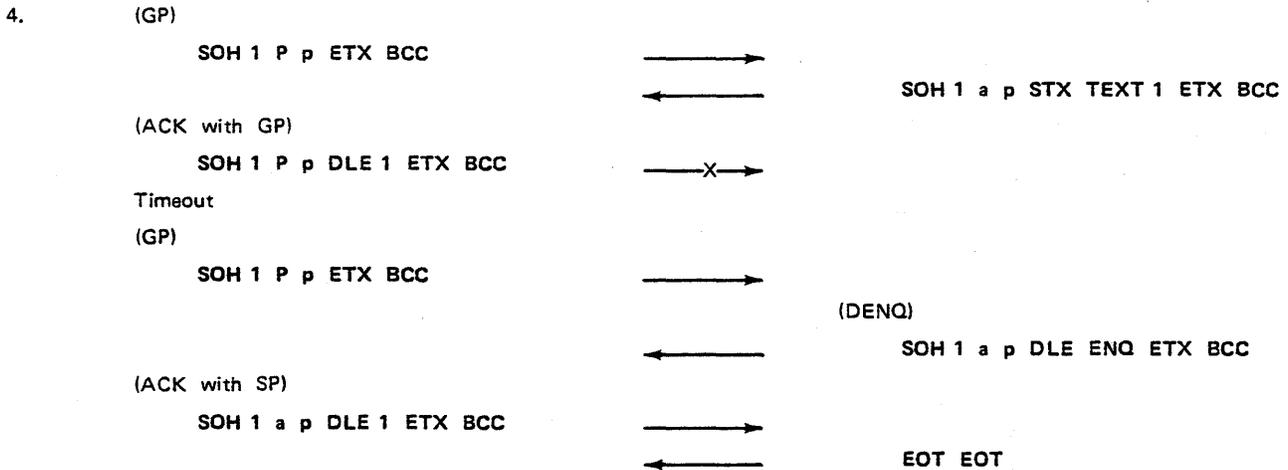
Example 1 is the normal case. The following examples are error cases.

2.	(GP) SOH 1 P p ETX BCC	→	
	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	←	SOH 1 a p STX TEXT 1 ETX BCC
	Timeout (GP) SOH 1 P p ETX BCC	← X	EOT EOT
		→	EOT EOT or SOH 1 a p STX TEXT 2 ETX BCC

If the UNISCOPE terminal did not use the EOT, the processor could not poll again, in this example, but would send DNAK, on the assumption that the hit was not on the ACK.



In example 3, the DNAK is used because there never was an ACK to be repeated.



In example 4, because of the use of EOT, the previous response to a terminal can be used to determine the proper response to a timeout situation.

Another consideration in error recovery is that the poll sent with the ACK should be a specific poll during recovery. The UNISCOPE terminal will respond to a general poll at this point in the procedure, however.

In example 3, DNAK was sent because no other response had been sent to terminal a previously. In example 4, there was a response to the terminal: an ACK had been sent in response to TEXT 1. Example 4 is the only case that calls for an ACK with GP to be sent in response to the DENQ. The principle governing these examples may be expressed as follows: The processor response to a DENQ from a terminal to which an ACK has just been sent is another ACK if the first ACK was for text; the response will be a DNAK if the ACK was not for text. Response to a DENQ from a terminal other than the one to which an ACK has just been sent is always DNAK. This principle can be illustrated more completely with examples of two or more active terminals on a multiplexer. Example 5 is a normal situation and examples 6 and 7 contain error conditions.

Example	Processor Message	Direction of Transmission	Terminal Message
5.	(GP) SOH 1 P p ETX BCC	→ ←	SOH 1 a p STX TEXT 1 ETX BCC
	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	→ ←	SOH 1 b p STX TEXT 1 ETX BCC
	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	→ ←	SOH 1 a p STX TEXT 2 ETX BCC
	Etc.		
6.	(GP) SOH 1 P p ETX BCC	→ ←	SOH 1 a p STX TEXT 1 ETX BCC
	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	→ ← X	SOH 1 b p STX TEXT 1 ETX BCC
	(GP) SOH 1 P p ETX BCC	→ ←	(DENO) SOH 1 b p DLE ENQ ETX BCC
	(DNAK) SOH 1 b p DLE NAK ETX BCC	→ ←	SOH 1 b p STX TEXT 1 ETX BCC
	Etc.		
7.	(GP) SOH 1 P p ETX BCC	→ ←	SOH 1 a p STX TEXT 1 ETX BCC
	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	← X	
	Timeout		
	(GP) SOH 1 P p ETX BCC	→ ←	(DENO) SOH 1 a p DLE ENQ ETX BCC
	(ACK with SP) SOH 1 a p DLE 1 ETX BCC	→ ←	EOT EOT
	Etc.		

Examples 8 through 15 illustrate the use of the DNAK response following an ACK that was not in response to text. The status poll falls into this category. Examples 8 and 9 are normal cases without parity errors; the other examples are error cases.

Example	Processor Message	Direction of Transmission	Terminal Message
8.	(General status poll)		
	SOH 1 P p ENQ ETX BCC	→	(Traffic)
		←	SOH 1 a p DLE 0 ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→	
		←	SOH 1 a p STX TEXT 1 ETX BCC
	Etc.		
9.			
	SOH 1 a p STX TEXT ETX BCC	→	
	(General status poll)		
	SOH 1 P p ENQ ETX BCC	→	
		←	(Traffic response from terminal b with ACK from terminal a)
			SOH 1 b p DLE 1 DLE 0 ETX BCC
	(ACK with general status poll)		
	SOH 1 P p DLE 1 ENQ ETX BCC	→	
		←	(Traffic response from either terminal a or terminal b)
			SOH 1 a (or b) p DLE 0 ETX BCC
	Etc.		
10.	(General status poll)		
	SOH 1 P p ENQ ETX BCC	→	
		←	(Traffic response)
			SOH 1 a p DLE 0 ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→ X →	
	Timeout		
	(GP)		
	SOH 1 P p ETX BCC	→	
		←	(DENO)
			SOH 1 a p DLE ENQ ETX BCC
	(DNAK)		
	SOH 1 a p DLE NAK ETX BCC	→	
		←	(Traffic response)
			SOH 1 a p DLE 0 ETX BCC
	Etc.		

Example	Processor Message	Direction of Transmission	Terminal Message
11.	(General status poll)		
	SOH 1 P p ENQ ETX BCC	→	
		←	(Traffic response)
			SOH 1 a p DLE 0 ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→	
		← X	SOH 1 a p STX TEXT 1 ETX BCC
	Timeout		
	(GP)		
	SOH 1 P p ETX BCC	→	
		←	(DENO)
			SOH 1 a p DLE ENQ ETX BCC
	(DNAK)		
	SOH 1 a p DLE NAK ETX BCC	→	
		←	SOH 1 a p STX TEXT 1 ETX BCC
	Etc.		
12.	SOH 1 a p STX TEXT ETX BCC	→	
	(General status poll)		
	SOH 1 P p ENQ ETX BCC	→	
		← X	(Traffic response from terminal b with ACK from terminal a)
			SOH 1 b p DLE 1 DLE 0 ETX BCC
	Timeout		
	(General status poll)		
	SOH 1 P p ENQ ETX BCC	→	
		←	(DENO)
			SOH 1 b p DLE ENQ ETX BCC
	(DNAK)		
	SOH 1 b p DLE NAK ETX BCC	→	
		←	(Traffic response from terminal b with ACK from terminal a)
			SOH 1 b p DLE 1 DLE 0 ETX BCC
	Etc.		
13.	SOH 1 a p STX TEXT ETX BCC	→	
	(General status poll)		
	SOH 1 P p ENQ ETX BCC	→	
		←	(Traffic response from terminal a with ACK)
			SOH 1 a p DLE 1 DLE 0 ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→	
		← X	SOH 1 a p STX TEXT 1 ETX BCC

Example	Processor Message	Direction of Transmission	Terminal Message
	Timeout (GP) SOH 1 P p ETX BCC	→	(DENQ)
	(DNAK) SOH 1 a p DLE NAK ETX BCC	←	SOH 1 a p DLE ENQ ETX BCC
	Etc.	→	SOH 1 a p STX TEXT 1 ETX BCC
14.	SOH 1 a p STX TEXT ETX BCC (GP) SOH 1 P p ETX BCC	→	
	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	→	(No traffic with ACK)
		←	SOH 1 a p DLE 1 ETX BCC
	Timeout (GP) SOH 1 P p ETX BCC	→	
	(DNAK) SOH 1 b p DLE NAK ETX BCC	←	(DENQ)
	Etc.	→	SOH 1 b p DLE ENQ ETX BCC
		←	SOH 1 b p STX TEXT 1 ETX BCC
15.	SOH 1 a p STX TEXT ETX BCC (GP) SOH 1 P p ETX BCC	→	
	(ACK with GP) SOH 1 P p DLE 1 ETX BCC	→	(No traffic with ACK)
	Timeout (GP) SOH 1 P p ETX BCC	→	
		←	(DENQ)
		←	SOH 1 a p DLE ENQ ETX BCC

Example	Processor Message	Direction of Transmission	Terminal Message
	(DPAK)		
	SOH 1 a p DLE NAK ETX BCC	→	
		←	(No traffic with ACK)
			SOH 1 a p DLE 1 ETX BCC
	Etc.		

Example 15 is the same as example 7 except that the processor ACK is for something other than TEXT. As a result, DPAK is the response to DENQ.

The following examples further illustrate the use of error recovery. Examples 16, 20, and 24 are normal cases; the other examples are error cases.

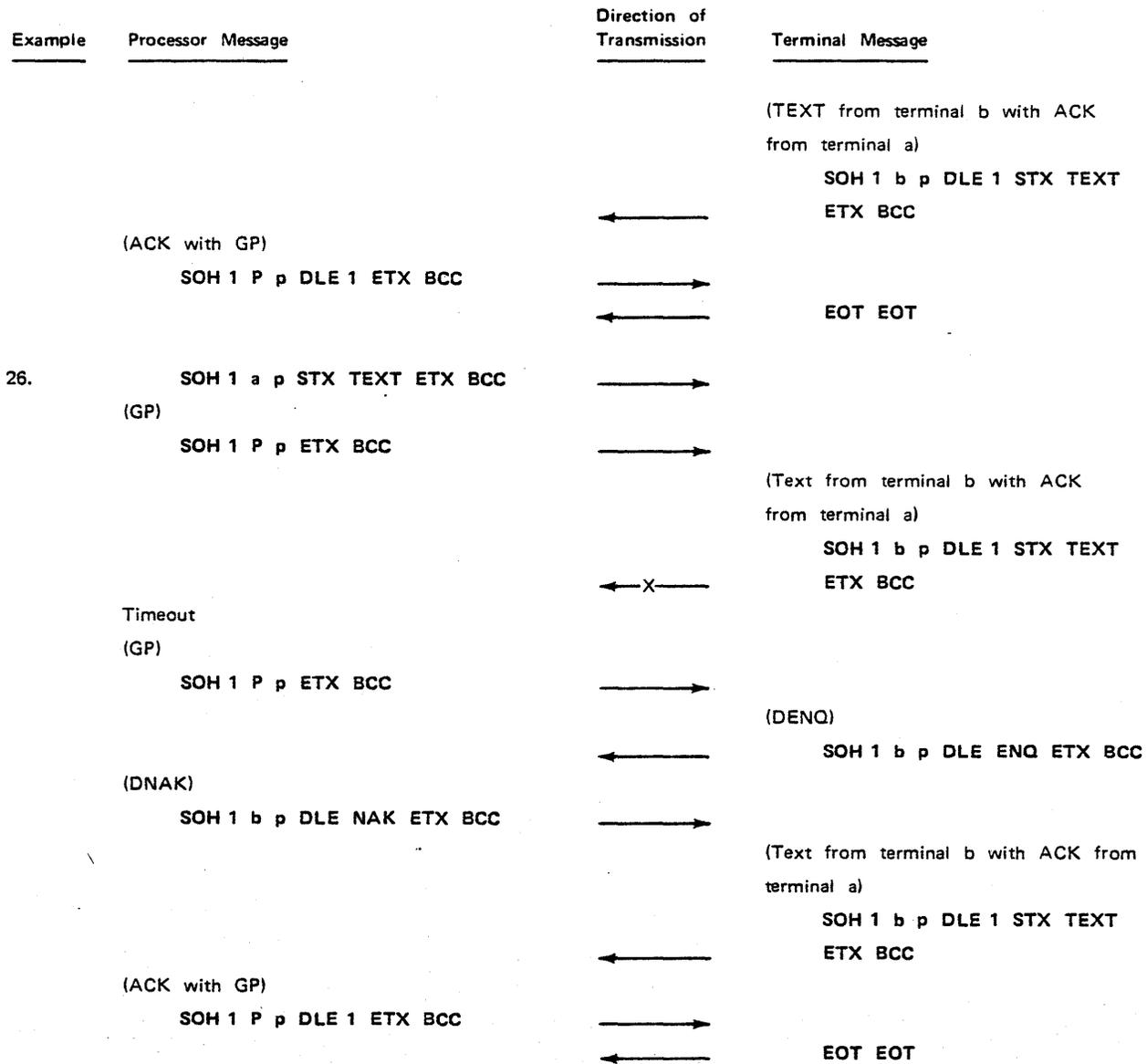
16.	(GP)		
	SOH 1 P p ETX BCC	→	
		←	SOH 1 a p STX TEXT ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→	
		←	SOH 1 b p STX TEXT ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→	
		←	(No traffic)
			EOT EOT
17.	(GP)		
	SOH 1 P p ETX BCC	→X	
	Timeout		
	(GP)		
	SOH 1 P p ETX BCC	→	
		←	SOH 1 a p STX TEXT ETX BCC
	Etc.		
18.	(GP)		
	SOH 1 P p ETX BCC	→	
		←	SOH 1 a p STX TEXT ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→X	
	Timeout		
	(GP)		
	SOH 1 P p ETX BCC	→	
		←	(DENQ)
			SOH 1 a p DLE ENQ ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→	
		←	SOH 1 b p STX TEXT ETX BCC
	Etc.		

Example	Processor Message	Direction of Transmission	Terminal Message
19.	(GP)		
	SOH 1 P p ETX BCC	→	
		←	SOH 1 a p STX TEXT ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→	
		←	SOH 1 b p STX TEXT ETX BCC
	(ACK with GP)		
	SOH 1 P p DLE 1 ETX BCC	→	
	← X	EOT EOT	
(GP)			
SOH 1 P p ETX BCC	→		
	←	EOT EOT	

The second EOT will not be present if a terminal has entered the traffic condition during recovery.

20.	SOH 1 a p STX TEXT ETX BCC	→	
	(SP)		
	SOH 1 a p ETX BCC	→	
		←	(No traffic with ACK)
	(ACK with SP)		SOH 1 a p DLE 1 ETX BCC
	SOH 1 a p DLE 1 ETX BCC	→	
	←	EOT EOT	
21.	SOH 1 a p STX TEXT 1 ETX BCC	→ X	
	(SP)		
	SOH 1 a p ETX BCC	→	
		←	EOT EOT
	SOH 1 a p STX TEXT 1 ETX BCC	→	
	(SP)		
	SOH 1 a p ETX BCC	→	
		←	(ACK with no traffic)
		SOH 1 a p DLE 1 ETX BCC	
(ACK with SP)			
SOH 1 a p DLE 1 ETX BCC	→		
	←	EOT EOT	
22.	SOH 1 a p STX TEXT ETX BCC	→	
	(SP)		
	SOH 1 a p ETX BCC	→ X	
	Timeout		
(SP)			
SOH 1 a p ETX BCC	→		

Example	Processor Message	Direction of Transmission	Terminal Message
			(No traffic with ACK)
		←	SOH 1 a p DLE 1 ETX BCC
	(ACK with SP)	→	
	SOH 1 a p DLE 1 ETX BCC	←	EOT EOT
23.	SOH 1 a p STX TEXT ETX BCC	→	
	(SP)	→	
	SOH 1 a p ETX BCC	→	
	Timeout	← X	(No traffic with ACK)
	(SP)		SOH 1 a p DLE 1 ETX BCC
	SOH 1 a p ETX BCC	→	
	(DNAK)	←	(DENQ)
	SOH 1 a p DLE NAK ETX BCC	→	SOH 1 a p DLE ENQ ETX BCC
	(ACK with SP)	←	(No traffic with ACK)
	SOH 1 a p DLE 1 ETX BCC	→	SOH 1 a p DLE 1 ETX BCC
		←	EOT EOT
24.	SOH 1 a p STX TEXT ETX BCC	→	
	(GP)	→	
	SOH 1 P p ETX BCC	→	
	(ACK with GP)	←	(TEXT from terminal b with ACK from terminal a)
	SOH 1 P p DLE 1 ETX BCC	→	SOH 1 b p DLE 1 STX TEXT ETX BCC
		←	EOT EOT
25.	SOH 1 a p STX TEXT ETX BCC	→	
	(GP)	→	
	SOH 1 P p ETX BCC	← X	
	Timeout		
	(GP)	→	
	SOH 1 P p ETX BCC	→	



### D.3. ERROR RECOVERY WITH AUXILIARY INTERFACE DEVICES

When dealing with auxiliary interface devices, the operations to be concerned with are selection, data transfer, and status reporting. Errors encountered may result from hits on the communications line or error conditions in the devices. In the following examples, proper error recovery procedures for a variety of conditions are illustrated. In the examples and accompanying text, the following terms are used for brevity:

- ACK — Acknowledge (DLE 1)
- DNAK — Retransmission request
- DS — Device status ready (DLE >)
- PR — Print command
- WABT — Auxiliary interface active (DLE ?)
- THRU — Auxiliary interface activity producing a WABT has been concluded (DLE ;)
- SP — Specific traffic poll
- Poll — Any type of poll (general or specific, traffic or status) where the response will always be the one shown.

### D.3.1. Errors Resulting From Hits on Communications Line

Error-producing hits on the communications line may occur during a poll, a poll response, or a text message. In determining the correct recovery sequence, it must first be determined where the error took place which requires determining whether:

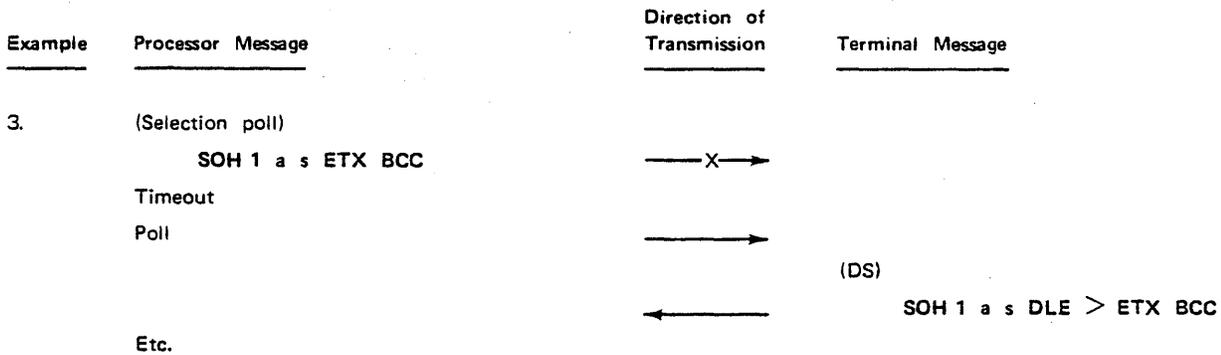
1. a selection has taken place;
2. the text was received correctly; or
3. an auxiliary interface transfer is taking place or has occurred.

#### D.3.1.1. Selection Poll

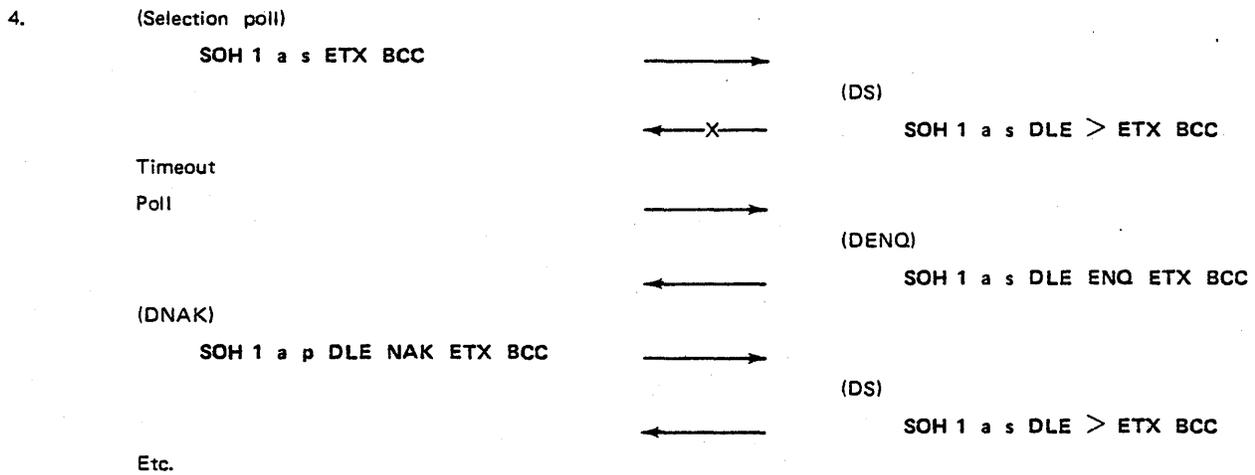
In the following, example 1 is a normal condition and examples 2, 3, and 4 are error conditions.

Example	Processor Message	Direction of Transmission	Terminal Message
1.	(Selection poll)		
	SOH 1 a s ETX BCC	→	(DS)
		←	SOH 1 a s DLE > ETX BCC
	Poll with ACK	→	(No traffic)
		←	EOT EOT
	(Text with Print Command)		
	SOH 1 a p STX TEXT DC2 ETX BCC	→	
	Poll	→	(WABT)
		←	SOH 1 a s DLE ? ETX BCC
	Poll with ACK	→	(THRU)
	←	SOH 1 a s DLE ; ETX BCC	
	→	(No traffic)	
	←	EOT EOT	
Etc.			
2.	(Selection poll)		
	SOH 1 a s ETX BCC	→ X →	
	Timeout		
	Poll	→	(No traffic)
		←	EOT EOT
	(Selection poll)		
SOH 1 a s ETX BCC	→		
Etc.			

In example 2, the lack of an acknowledgeable message from the terminal indicates that the error in the output message occurred prior to the DID and no selection attempt took place.

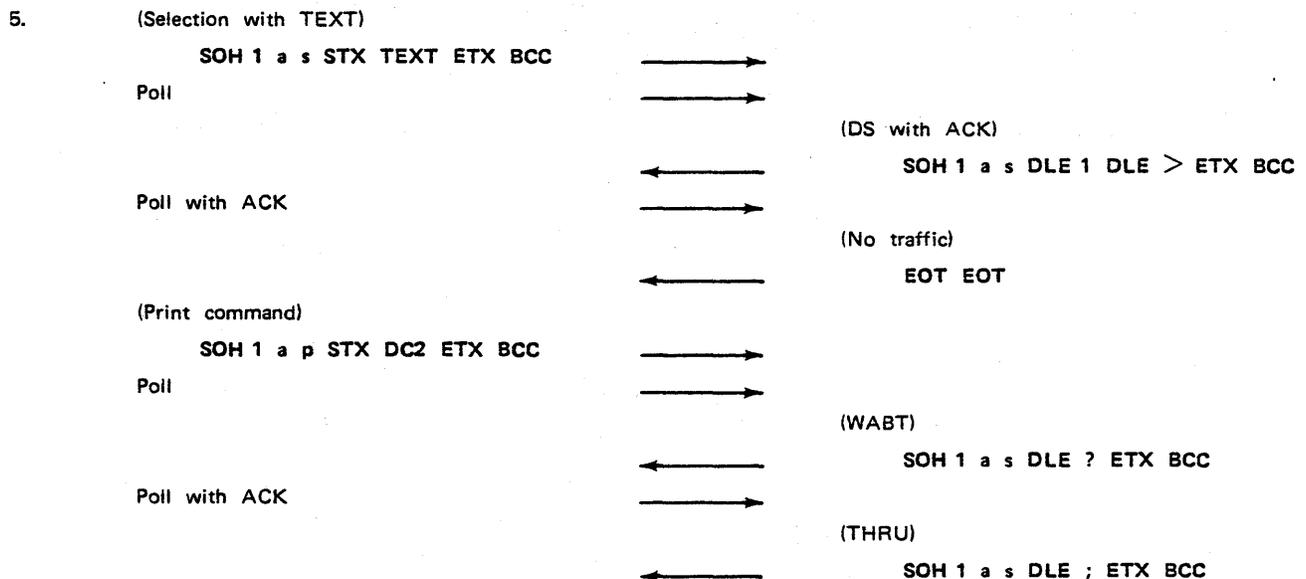


In example 3, the DS and the lack of an ACK indicate that the error occurred after the selection.



### D.3.1.2. Selection With Text

Example 5 is a normal condition and examples 6 through 12 are error conditions.



Example	Processor Message	Direction of Transmission	Terminal Message
	Poll with ACK	→	(No traffic)
		←	EOT EOT
	Etc.		
6.	(Selection with text)		
	SOH 1 a s STX TEXT 1 ETX BCC	→ X	
	Poll	→	
		←	EOT EOT
	(Selection with text)		
	SOH 1 a s STX TEXT 1 ETX BCC	→	
	Etc.		

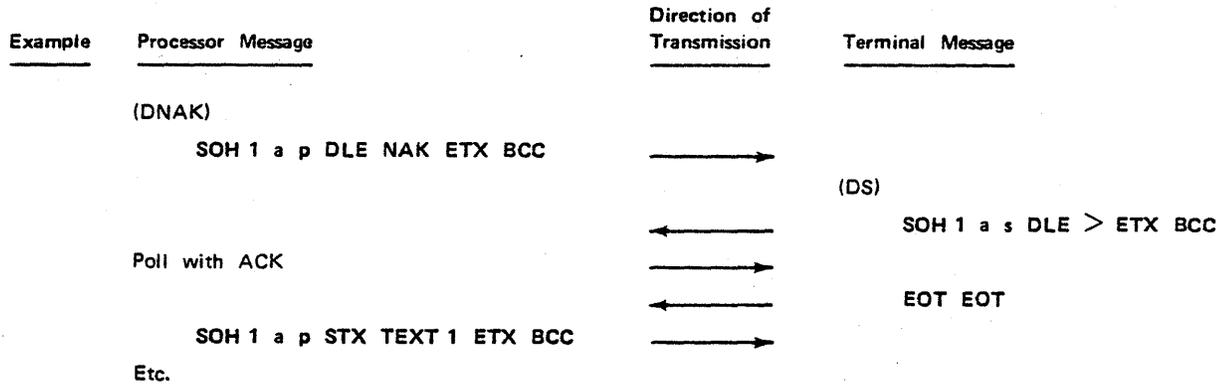
In example 6, the lack of an acknowledgeable message from the terminal indicates that the error in the output message occurred prior to the DID and no selection attempt took place.

7.	(Selection with text)		
	SOH 1 a s STX TEXT 1 ETX BCC	→ X	
	Poll	→	
		←	(DS)
		←	SOH 1 a s DLE > ETX BCC
	Poll with ACK	→	
		←	(No traffic)
		←	EOT EOT
	SOH 1 a p STX TEXT 1 ETX BCC	→	
	Poll	→	
		←	SOH 1 a s DLE 1 ETX BCC
	Poll with ACK	→	
		←	EOT EOT
	(Print command)		
	SOH 1 a p STX DC2 ETX BCC	→	
	Etc.		

In example 7, the DS and the lack of an ACK indicates that the error occurred after the selection.

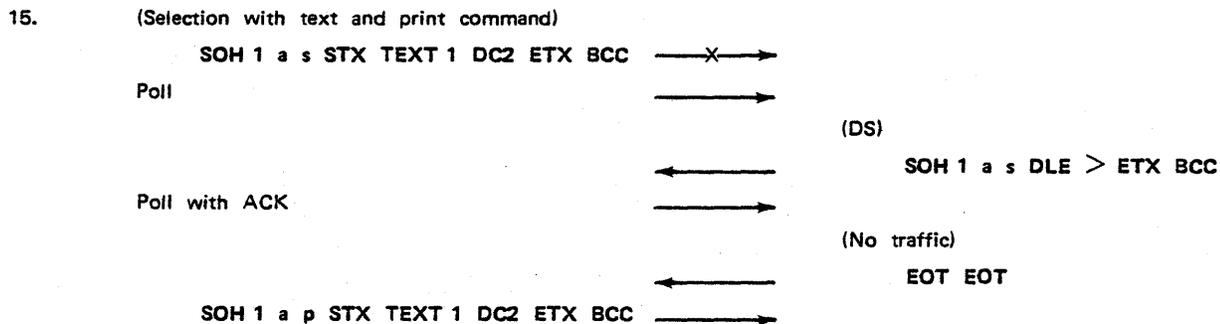
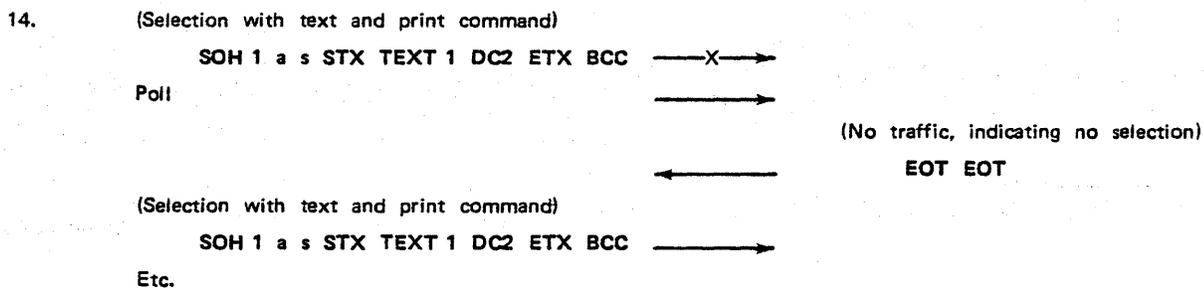
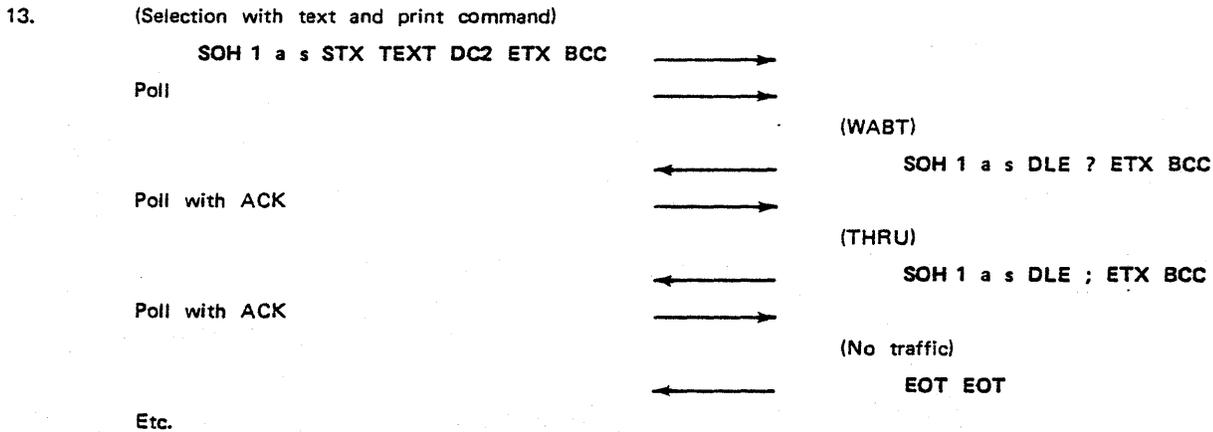
8.	(Selection with text)		
	SOH 1 a s STX TEXT 1 ETX BCC	→	
	Poll	→ X	
	Timeout		
	Poll	→	
		←	(DS with ACK)
		←	SOH 1 a s DLE 1 DLE > ETX BCC
	Etc.		

Example	Processor Message	Direction of Transmission	Terminal Message
9.	(Selection with text)		
	SOH 1 a s STX TEXT 1 ETX BCC	→	
	Poll	→	
		← X	(DS with ACK) SOH 1 a s DLE 1 DLE > ETX BCC
	Timeout		
	Poll	→	
		←	(DENQ) SOH 1 a s DLE ENQ ETX BCC
	(DPAK)		
	SOH 1 a p DLE NAK ETX BCC	→	
		←	(DS with ACK) SOH 1 a s DLE 1 DLE > ETX BCC
	Etc.		
10.	(Selection with text)		
	SOH 1 a s STX TEXT 1 ETX BCC	→ X	
	Poll	→ X	
	Timeout		
	Poll	→	
		←	(DS or No traffic, depending on what place in the message the error occurred)
	Etc.		
11.	(Selection with text)		
	SOH 1 a s STX TEXT 1 ETX BCC	→ X	
	Poll	→	
		← X	(No traffic) EOT EOT
	Timeout		
	Poll	→	
		←	EOT EOT
	(Selection with text)		
	SOH 1 a s STX TEXT 1 ETX BCC	→	
	Etc.		
12.	(Selection with text)		
	SOH 1 a s STX TEXT 1 ETX BCC	→ X	
	Poll	→	
		← X	(DS) SOH 1 a s DLE > ETX BCC
	Timeout		
	Poll	→	
		←	(DENQ) SOH 1 a s DLE ENQ ETX BCC



### D.3.1.3. Selection With Text and Print Command

Example 13 is a normal condition and examples 14 through 20 are error conditions.



Example	Processor Message	Direction of Transmission	Terminal Message
	Poll	→	(WABT)
	Etc.	←	SOH 1 a s DLE ? ETX BCC
16.	(Selection with text and print command)		
	SOH 1 a s STX TEXT 1 DC2 ETX BCC	→	
	Poll	→ X	
	Timeout		
	Poll	→	
	Etc.	←	(WABT) SOH 1 a s DLE ? ETX BCC
17.	(Selection with text and print command)		
	SOH 1 a s STX TEXT 1 DC2 ETX BCC	→	
	Poll	→	
	Timeout	← X	(WABT) SOH 1 a s DLE ? ETX BCC
	Poll	→	
	(DENAQ)	←	(DENQ) SOH 1 a s DLE ENQ ETX BCC
	SOH 1 a p DLE NAK ETX BCC	→	
	Etc.	←	(WABT) SOH 1 a s DLE ? ETX BCC
18.	(Selection with text and print command)		
	SOH 1 a s STX TEXT 1 DC2 ETX BCC	→ X	
	Poll	→ X	
	Timeout		
	Poll	→	
	Etc.	←	(No traffic or DS, depending on where the first error occurred)
19.	(Selection with text and print command)		
	SOH 1 a s STX TEXT 1 DC2 ETX BCC	→ X	
	Poll	→	
	Etc.	← X	EOT EOT

Example	Processor Message	Direction of Transmission	Terminal Message
	Timeout		
	Poll	→	
		←	EOT EOT
	(Selection with text and print command)		
	SOH 1 a s STX TEXT 1 DC2 ETX BCC	→	
	Etc.		
20.	(Selection with text and print command)		
	SOH 1 a s STX TEXT 1 DC2 ETX BCC	→ X	
	Poll	→	
		← X	(DS) SOH 1 a s DLE > ETX BCC
	Timeout		
	Poll	→	
		←	(DENQ) SOH 1 a s DLE ENQ ETX BCC
	(DNAK)		
	SOH 1 a p DLE NAK ETX BCC	→	
		←	(DS) SOH 1 a s DLE > ETX BCC
	Poll with ACK	→	
		←	EOT EOT
	(Text and print command)		
	SOH 1 a p STX TEXT 1 DC2 ETX BCC	→	
	Etc.		

### D.3.1.4. Auxiliary Interface Input Transfer

Example 21 is a normal condition and examples 22 through 25 are error conditions.

21.	(Selection poll)		
	SOH 1 a s ETX BCC	→	
		←	(DS) SOH 1 a s DLE > ETX BCC
	Poll with ACK	→	
		←	EOT EOT
	(Print command)		
	SOH 1 a p STX DC2 ETX BCC	→	
	Poll	→	
		←	(WABT) SOH 1 a s DLE ? ETX BCC
	Poll with ACK	→	
		←	(THRU) SOH 1 a s DLE ; ETX BCC

Example	Processor Message	Direction of Transmission	Terminal Message
	(Poll with ACK) SOH Poll DLE 1 ETX BCC	→ ←	EOT EOT
		⋮	
	Traffic poll	→ ←	Transmit condition imposed SOH 1 a s STX TEXT ETX BCC
	Poll with ACK	→ ←	EOT EOT
	Etc.		
22.	(Selection poll) SOH 1 a s ETX BCC	→	(DS)
		←	SOH 1 a s DLE > ETX BCC
	Poll with ACK	→ ←	EOT EOT
	(Print command) SOH 1 a p STX DC2 ETX BCC	→ ←	
	Poll	→ ←	EOT EOT
	(Print command) SOH 1 a p STX DC2 ETX BCC	→	
	Etc.		
23.	(Selection poll) SOH 1 a s ETX BCC	→	(DS)
		←	SOH 1 a s DLE > ETX BCC
	Poll with ACK	→ ←	EOT EOT
	(Print command) SOH 1 a p STX DC2 ETX BCC	→ ←	
	Poll	→ ←	(WABT)
		→ ←	SOH 1 a s DLE ? ETX BCC
	Timeout		
	Poll	→	(DENQ)
		←	SOH 1 a s DLE ENQ ETX BCC
	(DNAK) SOH 1 a p DLE NAK ETX BCC	→	(WABT)
		←	SOH 1 a s DLE ? ETX BCC
	Etc.		

Example	Processor Message	Direction of Transmission	Terminal Message
24.	(Selection poll)		
	SOH 1 a s ETX BCC	→	(DS)
		←	SOH 1 a s DLE > ETX BCC
	Poll with ACK	→	
		←	EOT EOT
	(Print command)		
	SOH 1 a p STX DC2 ETX BCC	→	
	Poll	→	(WABT)
		←	SOH 1 a s DLE ? ETX BCC
	Poll with ACK	→	
		←	(THRU)
		← X	SOH 1 a s DLE ; ETX BCC
	Timeout		
	Poll	→	(DENQ)
	←	SOH 1 a s DLE ENQ ETX BCC	
(DNAK)			
SOH 1 a p DLE NAK ETX BCC	→	(THRU)	
	←	SOH 1 a s DLE ; ETX BCC	
Etc.			
25.	(Selection poll)		
	SOH 1 a s ETX BCC	→	(DS)
		←	SOH 1 a s DLE > ETX BCC
	Poll with ACK	→	
		←	EOT EOT
	(Print command)		
	SOH 1 a p STX DC2 ETX BCC	→	
	SOH Poll ETX BCC	→	(WABT)
		←	SOH 1 a s DLE ? ETX BCC
	Poll with ACK	→	
		←	(THRU)
		←	SOH 1 a s DLE ; ETX BCC
	Poll with ACK	→	
		←	EOT EOT
		Transmit condition imposed	

<u>Example</u>	<u>Processor Message</u>	<u>Direction of Transmission</u>	<u>Terminal Message</u>
	Traffic poll	→	
	Timeout	←X→	SOH 1 a s STX TEXT 1 ETX BCC
	Traffic poll	→	
	(DLE)	←	(DENQ)
	SOH 1 a p DLE NAK ETX BCC	←	SOH 1 a s DLE ENQ ETX BCC
		→	
		←	SOH 1 a s STX TEXT 1 ETX BCC
	Etc.		

### D.3.2. Errors Resulting From Error Conditions in the Devices

Error conditions in the devices can result from conditions of normal equipment operation, from a temporary error condition in transfer activity, or as a result of hardware malfunction. The result of these conditions is failure to perform a correct selection, incomplete data transfer, or failure to perform an operation.

Error conditions resulting from normal equipment operation include:

1. A print-test or out-of-paper condition on a printer.
2. An end-of-tape condition on a tape cassette unit.
3. An open interlock condition (such as cover open).
4. Device power off.

Temporary error conditions in transfer activity include:

1. Timing error
2. Data error

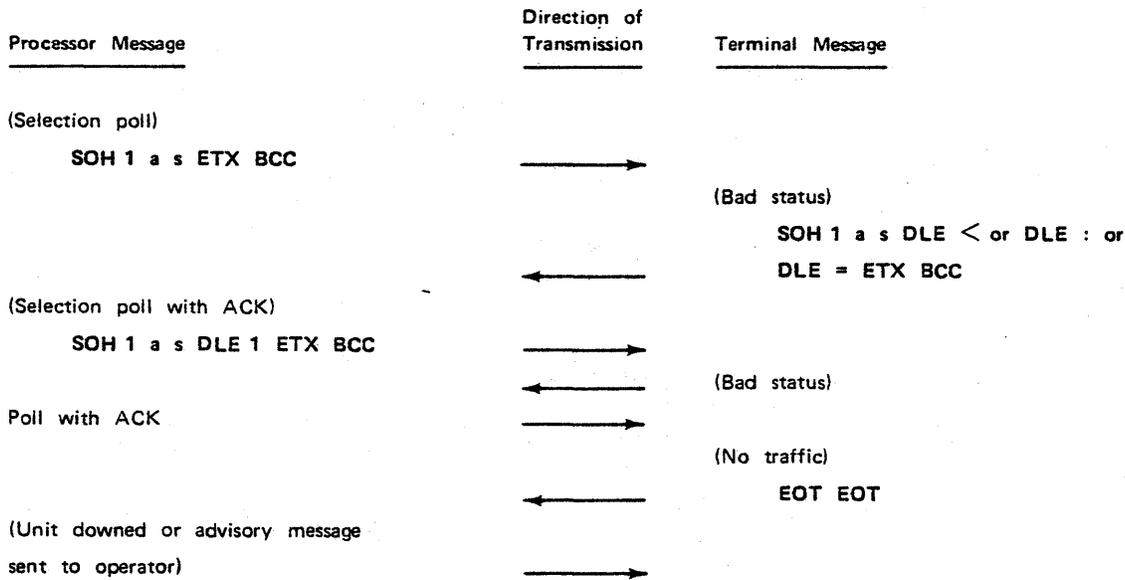
Hardware malfunctions causing errors include:

1. Fuse fault
2. Failure of a component

The recovery procedure depends on the type of error condition and the operation being attempted.

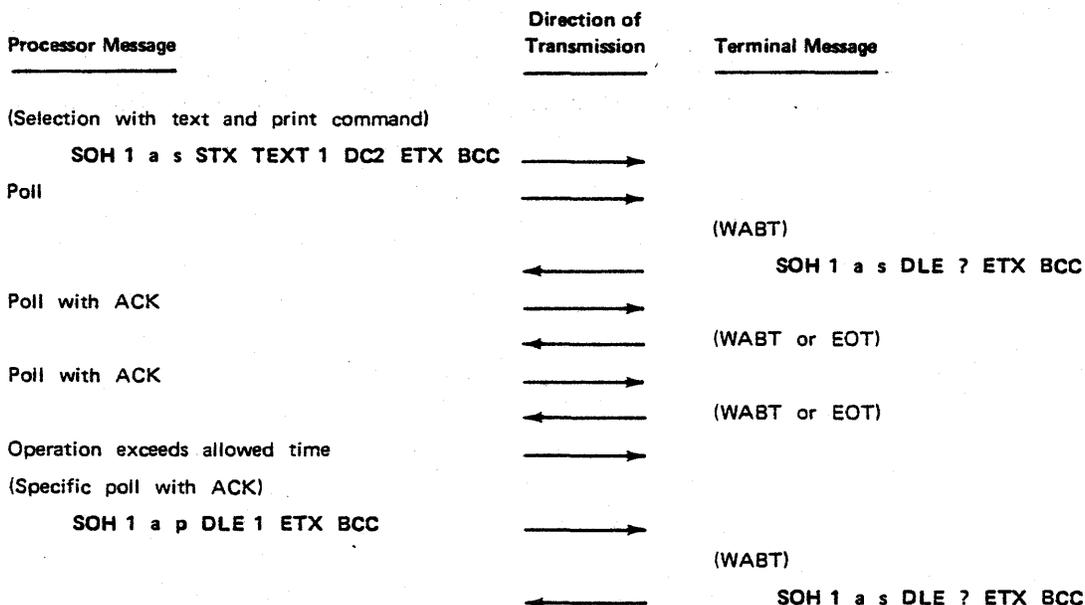
### D.3.2.1. Selection

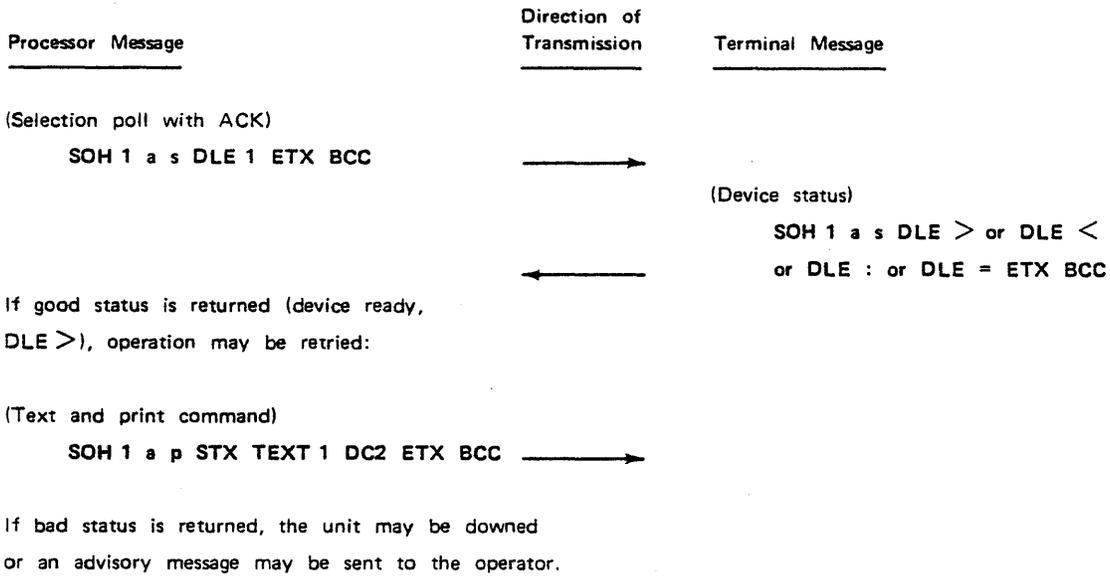
An error condition during an attempted selection will result in bad device status (status other than device ready). This error condition is indicated by a device selection status of DLE <, DLE :, or DLE =. If one of these responses is returned to a poll, the selection attempt may be repeated. If the selection status is still bad, the device or terminal may be temporarily downed (disregarded by the software as a unit in the communications network) or an advisory message may be sent to the operator. Following is an example of this type of error condition.



### D.3.2.2. Data Transfer or Operation

An error condition during a data transfer or operation normally results in a sustained busy condition. This is indicated by continuous WABT responses to specific polls or EOT responses to general polls and the absence of a THRU response. If this condition exceeds the time allowed for an operation, the device may be selected and its status solicited. The appropriate recovery will then be dependent on the device status. Following is an example of this type of error condition and the recovery procedure.





## Index

Term	Reference	Page	Term	Reference	Page
<b>A</b>					
Acknowledges			AID	See Auxiliary interface device	
alternating versus single	D.1.1	D-1	Alarm		
negative, error recovery	D.1.3	D-2	audible	2.2.7	2-10
special function section	2.2.3.6	2-8	cursor	2.2.2	2-4
Acknowledgments, line control	8.3	8-2	All-zeros character	3.6.3.1	3-29
Address, specific	3.4	3-10	Alphanumeric/symbolic section	2.2.3.1 Figure 2-3	2-5 2-5
Address, tape cassette system			Alternating acknowledges	D.1.1	D-1
illegal	4.6.2	4-18	American Standard Code for Information Interchange	See ASCII	
in search statement, online	4.6.1	4-16	ASCII		
recorded in block preamble	2.8.4.1	2-34	characters	2.1 2.2.3.1 Figure 2-4	2-1 2-5 2-6
reporting	2.8.9	2-39	coding, 7-bit	2.6 3.1	2-21 3-1
See also Report-address processor command search	See Search function		line-feed and form-feed functions	3.6.5.27	3-35
Address code assignments			octal code conversions	Table 2-3	2-23
multiplexer network	Figure 3-5	3-12	Asynchronous interface features		
one multiplexer divided into more than one logical multiplexer function	Figure 3-6	3-13	direct connection	2.4 Figure 2-5	2-13 2-14
primary/cascaded multiplexers	Figure 3-7	3-13	remote synchronous	Figure 2-6	2-15
single station	Figure 3-3	3-11			
terminals and multiplexers	Figure 3-4	3-12			
Addressing					
description	3.4	3-10			
first level	3.4.1	3-10			
second level	3.4.2	3-10			
third level	3.4.3	3-10			





Term	Reference	Page	Term	Reference	Page
Combined operation	2.6.2.3	2-28	Communications sequence (cont)		
Commands, processor control	4.5	4-12	description	2.5	2-19
Communications adapters			message		
typical system configuration	Figure 2-2	2-3	acknowledgment	2.5.4	2-20
2701 and SDAII	2.4.4	2-17	message format	2.5.3	2-19
2703 and Synchronous			Communications terminal		
Base 1	2.4.4	2-17	adapter, direct connection	2.4	2-13
Communications control			Communications Terminal		
characters			Module Controller	See CTMC	
BCC - block check			Configurations		
character	3.6.3.5	3-30	tape cassette system	2.8.2	2-33
DBR - break	3.6.2.3	3-28	terminal	1.3	1-2
DENQ - reply request	3.6.2.5	3-29		Figure 2-2	2-3
DLE - data link escape	3.6.1.7	3-28	Connection, direct	See Direct connection	
DNAK - negative			Connection through modem		
acknowledgment	3.6.2.2	3-28	substitutes	2.4	2-13
DRE - resume	3.6.2.4	3-29	Control and indicator panel	2.2.4	2-9
EOT - end of transmission	3.6.1.4	3-26	Control characters		
ENQ - inquiry	3.6.1.5	3-27	ASCII	Figure 2-4	2-6
ESC - escape	3.6.3.4	3-30	communications	See Communications	
ETX - end of text	3.6.1.3	3-26	corresponding octal		
miscellaneous	3.6.3	3-29	codes	Table 3-5	3-25
NUL	3.6.3.1	3-29	miscellaneous	3.6.3	3-29
SI - shift in	3.6.3.3	3-29	not placed in storage	Table 2-1	2-11
SO - shift out	3.6.3.2	3-29	Control code sequences	3.3.6	3-8
SOH - start of heading	3.6.1.1	3-26		Table 3-3	3-9
STX - start of text	3.6.1.2	3-26	Control codes		
supervisory messages	3.3.1	3-1	description	3.3.6	3-8
SYN - synchronous idle	Table 3-1	3-2	editing and device	Table 3-3	3-9
WABT - wait before	3.6.1.6	3-27	special function keys	3.3.3.2	3-6
transmit	3.6.2.1	3-28	Copy to address, tape cassette		
Communications interface between			system	2.8.1.2	2-32
UNISCOPE terminal and				4.9.5.3	4-23
processor	2.4	2-13	CR	3.6.5.1	3-31
Communication line errors			CTMC		
line control	B.1	B-1	2.1	2-1	
resulting from hits	D.3.1	D-14	2.4.1	2-13	
Communications sequence					
data flow, messages from					
terminal	2.5.2	2-19			
data flow, messages to					
terminal	2.5.1	2-19			

Term	Reference	Page	Term	Reference	Page
Cursor			Data format		
description	2.2.2	2-4	processor to terminal	3.5.2.1	3-16
processor-positioned	2.2.7.3	2-11	terminal to processor	Figure 3-11	3-17
wraparound	2.2.7.4	2-11		3.5.2.2	3-17
	2.7	2-29		Figure 3-12	3-18
	4.2.2	4-5	Data link escape (DLE)		
Cursor address			break and resume	3.3.2.3	3-4
description	3.5.1	3-14	description	3.6.1.7	3-28
screen display	Figure 3-10	3-15	DLE ENQ	See Reply request	
sequence	3.5	3-14	DLE NAK	3.3.2.1	3-3
Cursor control section	2.2.3.4	2-7	DLE 0	2.7	2-29
	Figure 2-3	2-5	DLE 8	2.6.2.4	2-28
Cursor move sequence, text format	3.5	3-14	Data retrieval (read), tape cassette system	See Read operation	
Cursor positioning sequence			Data rolling	3.5.3	3-18
initial	3.5.2.1	3-16	Data storage (write), tape cassette system	See Write operation	
text format	3.5	3-14	Data-terminal-ready signal	3.3.3.2	3-6
Cursor return			Data transfers		
character, description	3.6.5.1	3-31	combined operation	2.6.2.3	2-28
line feed	2.2.7.7	2-12	error conditions	D.3.2.2	D-24
sequence, text format	3.5	3-14	fast	D.3.2.2	D-24
text	3.5.2	3-15	online operation	2.6.2.1	2-24
Cursor to home			DBR	3.6.2.3	3-28
description	3.6.5.8	3-32	DCS	2.1	2-1
differences in units	2.7	2-29		2.4.1	2-13
text format	3.5	3-14	DCT 1000 Data Communications Terminal	1.1	1-1
			DC1 code	3.6.5.25	3-34
<b>D</b>			DC2 code	2.6.1.7	2-24
Data characters	3.6.4	3-31		3.6.5.23	3-34
Data Communications Subsystem	See SPERRY UNIVAC Data Communications Subsystem		DC4 code	2.7	2-29
Data fields, protected	3.5.7	3-24		3.6.5.22	3-34
Data flow			Delete character, DID register	2.6.1.3	2-22
messages from terminal	2.5.2	2-19			
messages to terminal	2.5.1	2-19			

Term	Reference	Page	Term	Reference	Page
Delete in display	3.6.5.4	3-31	Display		
Delete in line	3.6.5.5	3-32	erase	3.6.5.29	3-35
Delete line	3.6.5.16	3-33	processor-controlled	2.2.7	2-11
DENQ	3.6.2.5	3-29	regenerated	2.2.5	2-10
Delimiting character	3.6.3.5	3-30	transmit	3.6.5.28	3-35
Deselection conditions, tape cassette system	4.1.3 4.5.3 4.7 4.8.1 4.8.2 4.8.3	4-2 4-14 4-18 4-19 4-19 4-20	Display rolling		
Device control codes	See Editing and device control codes		description	2.2.7.5	2-12
Device identifier	See DID		roll down	3.5.3	3-18
Device-ready status (DLE > )	2.6.2.1	2-24	roll up	Figure 3-13	3-19
Device requirements	2.6.3	2-28	roll up	Figure 3-14	3-20
Devices, errors resulting from error conditions	D.3.2	D-23	Display screen, CRT	2.2.1	2-3
Dialed line	D.1.5	D-2	DLE	See Data link escape	
DID			DNAK	3.6.2.2	3-28
addressing	3.4.3	3-10	DRE	3.6.2.3	3-28
description	2.6.1.2	2-22	DTR	See Data-terminal-ready signal	
GID	3.4.4	3-10	Dummy block, function	4.2.2	4-5
message format	2.5.3	2-19			
register	2.6.1.3	2-22	<b>E</b>		
tape cassette system	2.8.2	2-33	Editing and device control codes		
Direct connection			BEL (processor) - message waiting	3.6.5.17	3-33
communications terminal adapter	2.4	2-13	BEL (terminal) - request processor message	3.6.5.18	3-33
configurations	Figure 2-5	2-14	CR - cursor return	3.6.5.1	3-31
description	2.4.1	2-13	DC1 - transmit	3.6.5.25	3-34
Direct interface	Figure 2-5	2-14	DC2 - print	3.6.5.23	3-34
			DC4 or ESC DC4 - lock keyboard	3.6.5.22	3-34
			ESC a - erase to end of display	3.6.5.2	3-31
			ESC b - erase to end of line	3.6.5.3	3-31
			ESC c - delete in line	3.6.5.5	3-32
			ESC C - delete in display	3.6.5.4	3-31
			ESC d - insert in line	3.6.5.7	3-32

Term	Reference	Page	Term	Reference	Page
Editing and device control codes (cont)			End-of-memory output	2.6.1.8	2-24
ESC D - insert in display	3.6.5.6	3-32	End-of-tape operation		
ESC DC1 - transmit display	3.6.5.28	3-35	functional description	2.8.10	2-40
ESC DC2 - print transparent	3.6.5.24	3-34	online procedures	4.7	4-18
ESC e - cursor to home	3.6.5.8	3-32	End-of-text	See ETX character	
ESC f - scan up	3.6.5.9	3-32	End of transmission	See EOT	
ESC g - scan left	3.6.5.10	3-32	ENQ	2.7	2-29
ESC h - scan right	3.6.5.11	3-32		3.6.1.5	3-27
ESC HT - tab stop set	3.6.5.19	3-34	Enquiry	2.7	2-29
ESC i - scan down	3.6.5.12	3-33		3.6.1.5	3-27
ESC j - insert line	3.6.5.15	3-33	Entry-edit capability	2.6	2-21
ESC k - delete line	3.6.5.16	3-33	EOMI		
ESC K - erase field	3.6.5.30	3-35	description	2.6.1.9	2-24
ESC M - erase display	3.6.5.29	3-35	online operation	2.6.2.1	2-24
FS - start-blink marker	3.6.5.20	3-34	EOMO	2.6.1.8	2-24
GS - end-blink marker	3.6.5.21	3-34	EOT	3.6.1.4	3-26
HT - tab	3.6.5.14	3-33	Equipment operation, error conditions	D.3.2	D-23
LF, FF - line feed and form feed	3.6.5.27	3-35	Erase display	2.7	2-29
RS - start of entry	3.6.5.26	3-35		3.6.5.29	3-35
SP - space	3.6.5.13	3-33	Erase field	3.6.5.30	3-35
Editing control section	2.2.3.5	2-8	Erase screen function, tape cassette system	2.8.4.5	2-36
	Figure 2-3	2-5	Erase to end of display	3.5.1	3-14
Editing keys				3.6.5.2	3-31
cursor	2.2.2	2-4	Erase to end of line	3.5.1	3-14
keyboard	Figure 2-3	2-5		3.6.5.3	3-31
Editing tape	2.8.1.2	2-32	Erasing tape		
	4.9.5.1	4-22	for reuse	4.6.2	4-18
EF	See External function		prevention	See Tape cassette, write protection	
EIA standard RS-232-C and CCITT	2.4.3	2-16	Erasure, characters	3.6.5.3	3-31
Embedded messages	3.3.2.3	3-4			
Embedded segment					
DBR	3.6.2.3	3-28			
DRE	3.6.2.4	3-29			
End-blink marker	3.6.5.21	3-34			
End-of-memory input	2.6.1.9	2-24			

Term	Reference	Page	Term	Reference	Page
Error conditions, temporary	D.3.2	D-23	Error status, tape cassette system		
Error detection, BCC	3.6.3.5	3-30	clearing, online	4.8.1	4-19
Error detection and recovery, tape cassette system				4.8.3	4-20
data errors, identification	2.8.6.1	2-37	reporting, online	4.1.2	4-2
error code, special	4.8	4-19		4.1.3	4-2
functional description	2.8.6	2-37	Errors, conditions in devices		
parity check	2.8.5	2-36	data transfer and operation	D.3.2.2	D-24
procedural errors	2.8.6.3	2-37	description	D.3.2	D-23
programming considerations	4.8	4-19	selection	D.3.2.1	D-24
read operation	4.8.1	4-19	ESC code	3.5.1	3-14
read-after-write operation	4.8.3	4-20		3.6.3.4	3-30
selection errors	4.8.4	4-20	ESC a code	3.6.5.2	3-31
tape problems	4.8.1	4-19	ESC b code	3.6.5.3	3-31
write operation	2.8.6.4	2-38	ESC c code	3.6.5.5	3-32
Error-free polls	B.2	B-1	ESC C code	3.6.5.4	3-31
Error-producing hits	D.3.1	D-14	ESC d code	3.6.5.7	3-32
Error recovery			ESC D code	3.6.5.6	3-32
after the timeout	D.1.6	D-2	ESC DC1 code	3.6.5.28	3-35
alternating versus single acknowledges	D.1.1	D-1	ESC DC2 code	2.6.1.10	2-24
auxiliary interface devices	D.3	D-13		3.6.5.24	3-34
auxiliary interface input transfer	D.3.1.4	D-20	ESC DC4 code	2.7	2-29
description	D.1	D-1		3.6.5.22	3-34
error conditions in devices	D.3.2	D-23	ESC e code	3.6.5.8	3-32
errors resulting from hits on communications line	D.3.1	D-14	ESC f code	3.6.5.9	3-32
examples	D.2	D-2	ESC g code	3.6.5.10	3-32
hardware rule 5 correctly used	D.2.2	D-4	ESC h code	3.6.5.11	3-32
hardware rule 5 not followed	D.2.1	D-3	ESC HT code	3.6.5.19	3-34
negative acknowledge	D.1.3	D-2	ESC i code	3.6.5.12	3-33
reply request	D.1.4	D-2	ESC j code	3.6.5.15	3-33
selection poll	D.3.1.1	D-14			
selection with text	D.3.1.2	D-15			
selection with text and print command	D.3.1.3	D-18			
timers	D.1.5	D-2			
two-character responses	D.1.2	D-1			



Term	Reference	Page	Term	Reference	Page
Horizontal tab	2.2.7.6	2-12	Insert line	2.2.7.3 3.6.5.15	2-11 3-33
HT	3.5.4 3.6.5.14	3-21 3-33	Interblock gap description invalid	2.8.4.1 4.6.2	2-34 4-18
HT translation character feature	2.8.1.2 4.9.2.3	2-32 4-22	Interface, auxiliary	See Auxiliary interface	
<b>I</b>			Interface channel, auxiliary	1.1	1-1
IBM System/360/370			Interface description, tape cassette system	2.8.4.3 to 2.8.4.5	2-35 to 2-36
remote asynchronous operation	2.4.5 Figure 2-9	2-18 2-18	Interface features	2.4	2-13
remote connection	2.4	2-13	Interface timing considerations	3.3.5 Table 3-2	3-7 3-8
remote synchronous operation	2.4.4 Figure 2-8	2-17 2-17	Interfaces, direct connection	Figure 2-5	2-14
Identifier search	2.8.1.2 4.9.4	2-32 4-22	Interrogation	2.5	2-19
See also Search function, modes B and C			<b>K</b>		
Identifiers, device (DIDs), tape cassette system	4.1	4-1	Keyboard		
IDR	2.6.1.1	2-22	alphanumeric/symbolic section	2.2.3.1	2-5
Illegal addresses	See Address, tape cassette system		cursor control section	2.2.3.4	2-7
Improper selection, tape cassette system	2.8.6.3 4.8.4	2-37 4-20	editing control section	2.2.3.5	2-8
Independent block check character	3.3.2.3	3-4	functions and octal codes	2.5.5 Table 2-2	2-21 2-21
Input data request	2.6.1.1	2-22	lock function	2.2.6 2.5.2	2-10 2-19
Input enable, auxiliary interface	2.6.1.4	2-22	message control section	2.2.3.3	2-7
Input transfer, auxiliary interface	D.3.1.4	D-20	numeric section	2.2.3.2	2-5
Insert in display	3.6.5.6	3-32	operation	2.2.3	2-4
Insert in line	3.6.5.7	3-32	protected format section	2.2.3.7	2-8
			special function section	2.2.3.6	2-8
			units with full protected format control	Figure 2-3	2-5

Term	Reference	Page	Term	Reference	Page
Keyboard lock code (DC4)			Message buffering	2.2.5	2-10
description	3.6.5.22	3-34	Message control section	2.2.3.3	2-7
differences in units	2.7	2-29	Figure 2-3		2-5
Keyed data	2.2.7	2-10	Message elements	3.2	3-1
<b>L</b>			Message formats		
LF	3.6.5.27	3-35	communications		
Line control rules			sequence	2.5.3	2-20
description	B.1	B-1	Figure 2-10		2-19
hardware rules	B.2	B-1	description	3.2	3-1
software rules	B.3	B-2	from terminal	Figure 3-2	3-5
traffic flow diagram	Figure B-1	B-3	to terminal	Figure 3-1	3-3
Line delete (ESC k)	2.2.7.4	2-11	Message sequences, tape		
	3.5.3	3-18	cassette system		
Line error, communication	B.1	B-1	automatic transmit	4.4	4-10
Line feed	2.2.7.7	2-12	backward one block	4.5.1	4-12
	3.6.5.27	3-35	online read	4.3.1	4-6
Line insert (ESC j)	2.2.7.3	2-11	online write	4.2.1	4-3
	3.5.3	3-18	report address	4.5.2	4-13
List function, tape cassette			search	4.5.3	4-14
system	2.8.1.2	2-32	selection	4.1.1	4-1
	4.9.5.1	4-22	Message types		
Load point	See Tape cassette		description	3.3	3-1
Lock conditions, online operation	2.6.2.1	2-24	from terminal	3.3.3	3-4
Lock keyboard	3.6.5.22	3-34	polls	3.3.3	3-4
Longitudinal parity	3.5.6	3-23	supervisory	3.3.1	3-1
	3.6.3.5	3-30	to terminal	3.3.2	3-2
<b>M</b>			urgent processor	3.3.4	3-7
Machine-sensible address	3.6.1.1	3-26	MESSAGE WAIT indicator	2.2.7.8	2-12
Media-fill function	3.6.3.1	3-29		3.6.5.17	3-33
Message acknowledgment	2.5.4	2-20		3.6.5.18	3-33
			Message waiting	3.6.5.17	3-33
			Messages		
			data flow, from terminal	2.5.2	2-19
			data flow, to terminal	2.5.1	2-19
			embedded	3.3.2.3	3-4
			missing	2.5.4	2-20
			request processor	3.6.5.18	3-33
			storing	2.2.5	2-10
			types	3.3	3-1
			unsolicited	2.2.7	2-10
				3.6.5.17	3-33
			urgent processor	3.3.4	3-7

Term	Reference	Page	Term	Reference	Page
Messages from terminal			Multistation with acknowledg-		
description	3.3.3	3-4	ment passed by multiplexer,		
supervisory	3.3.3.2	3-6	operation sequence example	C.5	C-5
text	3.3.3.1	3-4			
Messages to terminal			Multistation with two terminals		
break and resume	3.3.2.3	3-4	on a multiplexer, operation		
description	3.3.2	3-2	sequence example	C.4	C-4
polls	3.3.2.1	3-2			
text	3.3.2.2	3-4			
			<b>N</b>		
MIL-STD-188	2.4.3	2-16	Negative acknowledge		
Modem, synchronous	Figure 2-6	2-15	DPAK	3.6.2.2	3-28
Modem substitutes, connection	2.4	2-13	error recovery	D.1.3	D-2
Moving scroll	2.2.7.5	2-12	polls	3.3.2.1	3-2
Multidrop configuration	1.1	1-1	No traffic message	2.5	2-19
	2.6	2-21		3.3.3.2	3-6
Multiplexer address code			No-traffic-without-acknowledge		
assignments			response	3.3.3.2	3-6
divided into more than one			Nonselection polls	2.6.2.4	2-28
logical multiplexer			NUL	3.6.3.1	3-29
function	Figure 3-6	3-13	Numeric section	2.2.3.2	2-5
primary/cascaded	Figure 3-7	3-13		Figure 2-3	2-5
Multiplexer network, address					
code assignments	Figure 3-5	3-12	<b>O</b>		
Multiplexers, cascading	2.3	2-12	Octal/ASCII code conversions	Table 2-3	2-23
See also Terminal			Octal codes, keyboard		
multiplexers			functions	See Keyboard functions	
Multistation, two terminals with			and octal codes		
auxiliary interface devices on a			ODR	See Output data request	
multiplexer	C.6	C-7	Offline data transfer	2.6.2.3	2-28
Multistation configurations,			Offline operation	2.6.2	2-24
operating speeds	2.4.3	2-16		2.6.2.2	2-27
Multistation direct interface	Figure 2-5	2-14	Online operation		
Multistation synchronous/ asynchronous interface	Figure 2-6	2-15	description	2.6.2	2-24
				2.6.2.1	2-24
			status codes	Table 2-4	2-25

Term	Reference	Page
Operating speeds		
asynchronous operation	2.4.3	2-16
interface	2.6	2-21
synchronous operation	2.4.1	2-13
Operation		
combined	2.6.2.3	2-28
description	2.2	2-3
DLE 8	2.6.2.4	2-28
offline	2.6.2.2	2-27
online	2.6.2.1	2-24
processor-controlled	See Processor-controlled operation	
without AI	2.6.4	2-29
Operation sequence examples		
description	C.1	C-1
multistation, two terminals with auxiliary interface devices on a multiplexer	C.6	C-7
multistation with acknowledgment passed by multiplexer	C.5	C-5
multistation with two terminals on a multiplexer	C.4	C-4
single station	C.2	C-1
single station with auxiliary interface device	C.3	C-2
Operator enquiry	2.2.7	2-10
Options, tape cassette system	2.8.1.2	2-32
Output, simulated	2.6.2.1	2-24
Output data request, auxiliary interface output enable	2.6.1.5	2-22
Output data stream, AI	2.6.2.1	2-24
Output enable, auxiliary interface	2.6.1.5	2-22
Overwriting	2.2	2-3

Term	Reference	Page
<b>P</b>		
Pad character	3.5.6	3-23
Parity		
BCC parity character coding	Table 3-4	3-23
character	3.5.5.1, 3.5.5.2	3-22
check, tape cassette system	2.8.5	2-36
	2.8.6.1	2-37
communications sequence	2.5.1	2-19
description	3.5.6	3-23
longitudinal	See Longitudinal parity	
message buffering	2.2.5	2-10
Poll responses, combined operation	2.6.2.3	2-28
Polling	2.5	2-19
Polling rates, slow	B.2	B-1
Polls		
description	3.3.2.1	3-2
error-free	B.2	B-1
formats	Figure 3-1	3-3
	Figure 3-2	3-5
messages to terminal	3.3.2	3-2
nonselection, DLE 8	2.6.2.4	2-28
offline operation	2.6.2.2	2-27
online operation	2.6.2.1	2-24
responses	See Responses to polls	
status	3.3.2.1	3-2
traffic	See Traffic polls	
Postamble, tape block	2.8.4.1	2-34
Preamble, tape block	2.8.4.1	2-34
Primary terminal multiplexers	3.4.1	3-10
Print, initiate auxiliary interface function (DC2)	3.6.5.23	3-34
Print code	2.6.1.7	2-24
Print command, selection with text	D.3.1.3	D-18

Term	Reference	Page	Term	Reference	Page
Print command, use with tape cassette system			Processor messages		
functional description	2.8.4.3	2-35	request	3.6.5.18	3-33
read	4.3.1	4-6	urgent	3.3.4	3-7
search	4.6.1	4-16	Processor polls, terminal responses	Appendix A	
write	4.2.1	4-3	Processor-positioned cursor	2.2.7.3	2-11
PRINT key	2.6.1.6	2-24		2.2.7.4	2-11
Print-transparent code			Processor-to-terminal message formats	3.3.3.1	3-4
description	2.6.1.10	2-24		Figure 3-1	3-3
ESC DC2	3.6.5.24	3-34	Processors		
online operation	2.6.2.1	2-24	interface with UNISCOPE terminal	2.4	2-13
Print-transparent feature, tape cassette system	2.8.1.2	2-32	SPERRY UNIVAC	See SPERRY UNIVAC processors	
Printer	1.1	1-1	Protected data fields	3.5.7	3-24
Printout, hard-copy	1.1	1-1	Protected format		
Priority	2.3	2-12	description	3.5.7	3-24
Private line	2.4	2-13	differences in units	2.7	2-29
Processor control commands, tape cassette system	4.5	4-12	example of protected information displayed on screen	Figure 3-18	3-24
Processor-controlled display	2.2.7	2-10	example of protected information transmitted from processor terminal	Figure 3-17	3-24
Processor-controlled operation			example of unprotected information added to protected format display	Figure 3-19	3-24
control characters not placed in storage	Table 2-1	2-11	example of unprotected information transmission from terminal to processor	Figure 3-20	3-25
cursor return/line feed	2.2.7.7	2-12	feature	2.2	2-3
description	2.2.7	2-10	keys	2.2.3	2-4
display rolling	2.2.7.5	2-12	option, tape cassette system	2.8.1.2	2-32
field blinking	2.2.7.2	2-11	programming considerations, tape cassette system	4.9.2	4-21
horizontal tab	2.2.7.6	2-12	section	2.2.3.7	2-8
line delete	2.2.7.4	2-11			
line insert	2.2.7.3	2-11			
MESSAGE WAIT indicator	2.2.7.8	2-12			
request retransmission	2.2.7.1	2-11			
Processor message waiting (BEL), online operation	2.6.2.1	2-24			
See also BEL					

Term	Reference	Page	Term	Reference	Page
<b>R</b>					
Read operation, tape cassette system			Reply request		
functional description	2.8.4.4	2-35	description	3.6.2.5	3-29
online, processor-controlled	4.3	4-5	error recovery	D.1.4	D-2
Read-after-write function			from processor	3.3.3.2	3-6
error reporting	2.8.6.5	2-38	polls	3.3.2.1	3-2
option	2.8.1.2	2-32	Report-address processor command		
programming			description	2.8.9	2-39
considerations	4.8.3	4-20	message sequence	4.5.2	4-13
Receive-only terminal	1.3	1-2	use	4.5	4-12
Receive-transmit terminals	1.3	1-2	Request processor message (BEL)		
Record separator writing option	2.8.1.2	2-32	description	3.6.5.18	3-33
programming			polls	3.3.2.1	3-2
considerations	4.9.3	4-22	Request retransmission	2.2.7.1	2-11
Recovery from communication line error	B.1	B-1	Resistor block, terminating	2.8.2	2-33
Remote asynchronous operation			Responses to polls		
IBM system/360/370	2.4.5	2-18	by terminal, either single station or on a multiplexer	Figure A-1	A-1
SPERRY UNIVAC processors	2.4.3	2-16	by terminal, on a multiplexer	Figure A-2	A-2
	Figure 2-9	2-18	by terminal, with auxiliary interface	Figure A-3	A-2
	Figure 2-7	2-16	by terminal, with auxiliary interface and on multiplexer description	Figure A-4 Appendix A	A-3
Remote connection			Resume	3.6.2.3	3-30
IBM system/360/370	2.4	2-13	Retransmission request (OLE NAK)		
SPERRY UNIVAC processors	2.4	2-13	description	2.2.7.1	2-11
Remote identifier	See RID		negative acknowledge	D.1.3	D-2
Remote synchronous operation			polls	3.3.2.1	3-2
IBM system/360/370	2.4.4	2-17	Rewind operation, online	2.8.7	2-38
SPERRY UNIVAC processors	2.4.2	2-14		4.6.2	4-18
	Figure 2-8	2-17	RID		
	Figure 2-6	2-15	addressing	3.4.1	3-10
Remote terminal device	1.1	1-1	GID	3.4.4	3-10
			message format	2.5.3	2-19
			SID	3.4.2	3-10

Term	Reference	Page	Term	Reference	Page
Roll down function	3.5.3 Figure 3-13	3-18 3-19	Serial number	2.7	2-29
Roll up function	3.5.3 Figure 3-14	3-18 3-20	Serial split	2.7	2-29
Routing	3.4	3-10	Shift in (SI)		
RS	3.6.5.26	3-35	description	3.6.3.3	3-29
			online operation	2.6.2.1	2-24
			Shift out (SO)		
			description	3.6.3.2	3-29
			online operation	2.6.2.1	2-24
			SI	See Shift in	
			SID		
Scan down	3.6.5.12	3-33	addressing	3.4.2	3-10
Scan left	3.6.5.10	3-32	GID	3.4.4	3-10
Scan right	3.6.5.11	3-32	message format	2.5.3	2-19
Scan up	3.6.5.9	3-32	Simulated output	2.6.2.1	2-24
Screen display, cursor address	Figure 3-10	3-15	Single acknowledges	D.1.1	D-1
Scroll effect	3.5.3	3-18	Single station		
Search function			direct interface	Figure 2-5	2-14
functional description	2.8.7	2-38	operation sequence		
online use			example	C.2	C-1
command formats	4.6	4-16	synchronous interface	Figure 2-6	2-15
message sequences	4.5.3	4-14	with auxiliary interface		
modes @, A, B, C	4.6	4-16	device, operation		
sequence example			sequence example	C.3	C-2
Selection			Single station configurations, address code assignments	Figure 3-3	3-11
description	2.6.1.1	2-22	SO	See Shift out	
errors resulting from error			SOE character		
conditions in devices	D.3.2.1	D-24	cursor	2.2.2	2-4
with text	D.3.1.2	D-15		2.8.4.5	2-36
with text and print			keyboard operation	2.2.3	2-4
command	D.3.1.3	D-18	message format	2.5.3	2-19
Selection, tape cassette system			PRINT key	2.6.1.6	2-24
functional description	2.8.4.2	2-35	RS code	3.6.5.26	3-35
message sequences	4.1.1.	4-1	Software rules, line control	B.3	B-2
processor control commands	4.5	4-12	SOH character		
reporting	4.1.2	4-2	description	3.6.1.1	3-26
Selection poll, errors resulting			message format	2.5.3	2-19
from hits on communications					
line	D.3.1.1	D-14			

Term	Reference	Page	Term	Reference	Page
Space (SP)	3.6.5.13	3-26	Status codes	Table 2-4	2-25
Space suppression, nonsignificant	3.6.5.25	3-34	Status information	3.3.2.1	3-2
Special function codes	2.2.3.6	2-8	Status polls		
	3.3.3.2	3-6	description	3.3.2	3-2
Special function keys	3.3.3.2	3-6	format	3.3.2.1	3-2
Special function section	2.2.3.6	2-8		Figure 3-1	3-3
	Figure 2-3	2-5	Status, tape cassette system		
Specific address	3.4	3-10	error	See Error status, tape cassette system	
Specific poll	2.3	2-12	reporting, online		
SPERRY UNIVAC Communications Terminal Module Controller (CTMC)	See CTMC		code		
SPERRY UNIVAC Data Communications Subsystem (DCS)	2.1	2-1	interpretation	4.1.3	4-2
	2.4.1	2-13	functional		
SPERRY UNIVAC processors			description	4.1.2	4-2
remote asynchronous	2.4.3	2-16	codes	Table 4-1	4-2
	Figure 2-7	2-16	Stencil	2.2	2-3
remote connection	2.4	2-13	STOP function, caution	4.7	4-18
	2.4.2	2-14	Storage medium	1.1	1-1
	Figure 2-6	2-15	Strap option, message format	2.5.3	2-20
SPERRY UNIVAC Terminal Multiplexer	1.1	1-1	Strapping, terminal	2.2.3	2-4
	2.1	2-1	STX character		
Start-blink marker	3.6.5.20	3-34	description	3.6.1.2	3-26
Start-of-entry character	See SOE character		message format	2.5.3	2-19
Start-of-heading character	2.5.3	2-19	SUB codes, DC1 (transmit)	3.6.5.25	3-34
	3.6.1.1	3-26	Supervisory messages		
Start-of-text character	2.5.3	2-20	description	3.3.1	3-1
	3.6.1.2	3-26	from terminal	3.3.3	3-4
Station identifier	See SID			3.3.3.2	3-6
Status, DLE 8	2.6.2.4	2-28	Supervisory sequence	2.5.4	2-20
			SYN character		
			data flow, messages to		
			terminal	2.5.1	2-19
			description	3.6.1.6	3-27
			message format	2.5.3	2-19

Term	Reference	Page
Synchronous base 1 communications adapters	2.4.4	2-17
Synchronous idle characters	See SYN	
Synchronous interface		
direct connection	2.4 Figure 2-5	2-13 2-14
remote asynchronous	2.4.3 Figure 2-7	2-16 2-16
remote synchronous	Figure 2-6	2-15
Synchronous modem	Figure 2-6	2-15
Synchronous operation		
direct connection	2.4.1	2-13
remote	See Remote synchronous operation	
Synchronous systems, responses to polls	Appendix A	
Synchronous transmission		
character structure	3.5.5.1 Figure 3-15	3-22 3-22
message format	2.5.3	2-20
System configuration	Figure 2-2	2-3

## T

Term	Reference	Page
interchangeability	2.8.3	2-33
load point, definition	2.8.11	2-40
problems	2.8.6.4 4.6.2 4.8.1	2-38 4-18 4-19
reuse	4.6.2	4-18
write protection	2.8.1	2-31
Tape cassette system		
auxiliary device	1.2	1-2
description	2.8.1	2-31
configuration	2.8.2	2-33
programmer information and considerations	Section 4	
types	2.8	2-30
Tape format	2.8.4.1	2-34
Tape erasure	See Erasing tape	
Tape position	See Address, tape cassette system	
Tape repositioning	See Backward-one-block function; End-of-tape operations; Rewind; Search	
Tape transport		
description	2.8.1	2-31
selecting for operation	4.1	4-1
Terminal acknowledgment	2.5.4	2-20
Terminal multiplexers		
connections	2.3	2-12
primary	3.4.1	3-10
SPERRY UNIVAC	1.1	1-1
	2.1	2-1
Terminal strapping	2.2.3	2-4
Terminal-to-processor response message formats	Figure 3-2	3-5
Terminal-to-processor supervisor messages	3.3.3.2	3-6
Terminal-to-processor text	3.3.3.1	3-4



Term	Reference	Page
UNISCOPE Display Terminal (cont)		
operation without AI	2.6.4	2-29
responses	Appendix A	
selections	Figure 1-2	1-2
terminal multiplexer		
connections	2.3	2-12
terminal responses to polls	Appendix A	
transmission sequences	2.3	2-12
text format	3.5	3-14
traffic flow diagram	Figure B-1	B-3
typical system configuration	Figure 2-2	2-3
Uqits, differences	2.7	2-29
Unlocking, automatic	2.2.6	2-10
Unprotected display, transmit	3.6.5.25	3-34
Unprotected format	2.2.3	2-4
Unsolicited message	2.2.7	2-10
	3.6.5.17	3-33
Urgent processor message	3.3.4	3-7

## W

WABT		
description	3.6.2.1	3-28
online operation	2.6.2.1	2-24
response from processor	3.3.3.2	3-6
Wait before transmit	See WABT	
Write operation		
functional description	2.8.4.3	2-35
online, processor control	4.2	4-3
Write protection	See Tape cassette	

Term	Reference	Page
------	-----------	------

## USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

*Please note: This form is not intended to be used as an order blank.*

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update No.)*

### Comments:

Cut along line.

**From:**

---

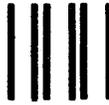
*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)  
Thank you for your cooperation

OLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

---

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

---

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY UNIVAC**

**ATTN.: SYSTEMS PUBLICATIONS**

P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19424



C

OLD