SPERRY UNIVAC
COMPUTER SYSTEMS

PUBLICATIONS UPDATE

General

SPERRY UNIVAC Universal Terminal System 400 Programmer Reference

UP-8359 Rev. 1-B

This library memo announces the release and availability of Update Package B for "SPERRY UNIVAC® Universal Terminal System 400 Programmer Reference," UP-8359 Revision 1. This is a Restricted Distribution Item (RD). Order where required.

Update Package B provides additional information on use of the SPERRY UNIVAC 0791 Correspondence Quality Printer Subsystem with the Universal Terminal System 400.

Copies of Update Package B are now available for requisitioning. Either the update package alone, or the complete manual including the update package, may be ordered by your Sperry Univac representative. To receive the update package alone, order UP-8359 Rev. 1-B. To receive the complete manual, including the update package, order UP-8359 Rev. 1.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Lists BZ, CZ (less DE, GZ, HA), MZ, 3, 8, 9, 9U, 31U, 37, 37U, 38, 41, 42, 43, 44, 51, 1D, 52, 53, 3D, 54, 4D, 55, 5D, 56, 57, 58, 60, 61, 62, 63, 63U, 64, 64U, 65, 66, 67, 68, 75, 76, 77, 78, 81, 81U, 83, 83U, 89, 89U	Lists DE, GZ, HA, ZZ, ZY, 82 (8 pages plus library memo)	Library Memo for UP-8359 Rev. 1-B RELEASE DATE: February 1981

SPERRY UNIVAC

Universal Terminal System 400

Programmer Reference

R

SPERRY UNIVAC

UP-8359 Rev. 1

SPERRY UNIVAC Universal Terminal System 400

Programmer Reference

PAGE STATUS SUMMARY
ISSUE: Update B – UP-8359 Rev. 1

Part/Section	Page Number	Update Level
Cover		Orig.
Title Page		B
PSS	1	B
Contents	1 thru 6 7, 8 8a 9 thru 11	Orig. A A* Orig.
1	1 2 thru 4 5 thru 7	Orig. A Orig.
2	1 thru 5 6 7, 8 9	Orig. A Orig. A
3 4	1 thru 63 1 thru 5 6 7 thru 22	Orig. Orig. A Orig.
5	1 thru 52	Orig.
Appendix A	1 thru 6	Orig.
Appendix B	1 thru 8	Orig.
Appendix C	1 thru 9	Orig.
Appendix D	1 2, 3 4 5 thru 7 8	A Orig. A B B*
Appendix E	1 thru 7	Orig.
Appendix F	1 thru 41	Orig.
Appendix G	1 thru 9	Orig.
Appendix H	1 thru 7	Orig.

Part/Section	Page Number	Update Level
Glossary	1 thru 5	Orig.
Index	1 thru 10	Orig.
User Comment Sheet		

Part/Section	Page Number	Update Level

**New Pages* → **New Pages*

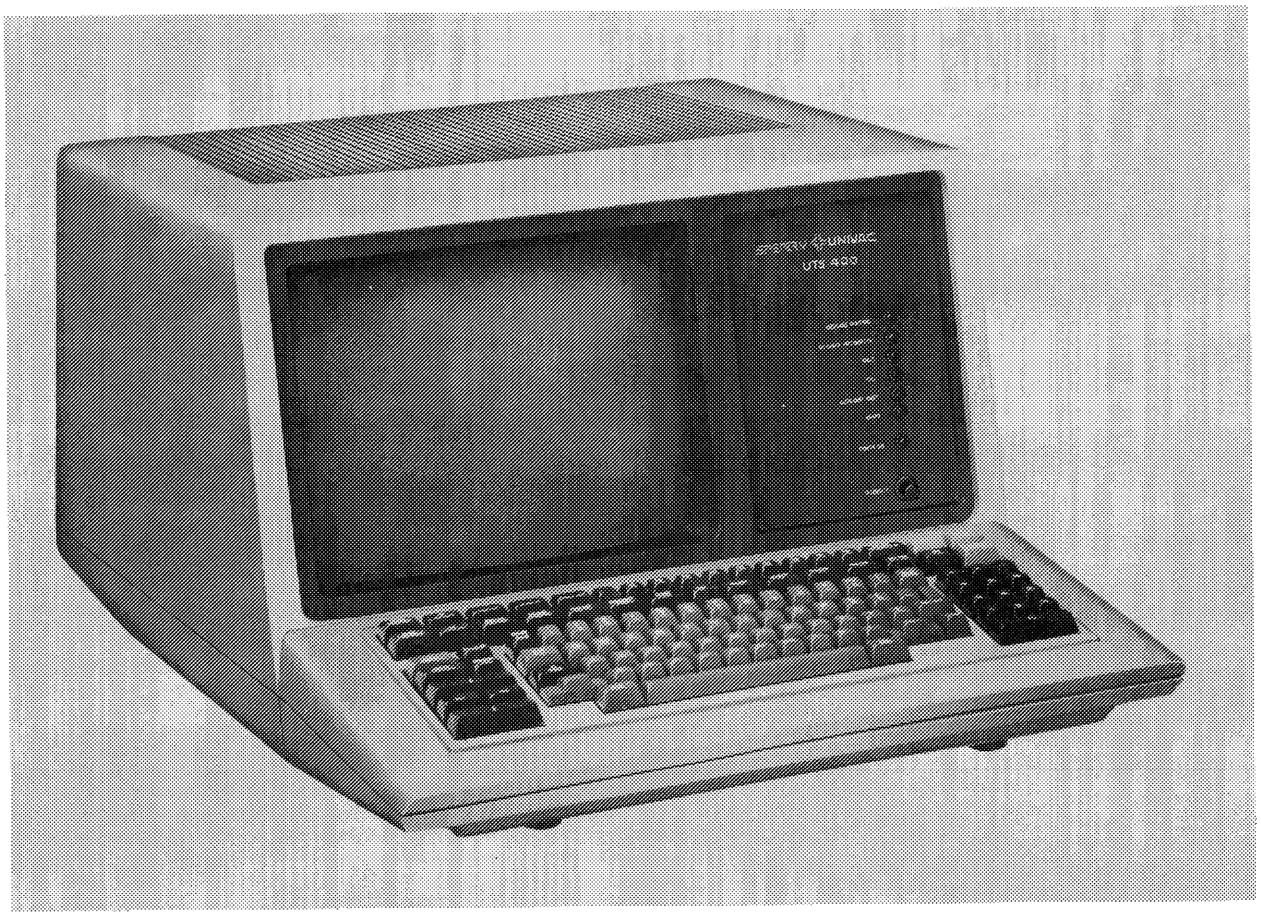
All the technical changes are denoted by an arrow (→) in the margin. A downward pointing arrow (↓) next to a line indicates that technical changes begin at this line and continue until an upward pointing arrow (↑) is found. A horizontal arrow (→) pointing to a line indicates a technical change in only that line. A horizontal arrow located between two consecutive lines indicates technical changes in both lines or deletions.

Contents

1. Introduction

1.1. GENERAL

This programmer reference manual contains information which must be considered in programming a central processing system that includes one or more units of the SPERRY UNIVAC Universal Terminal System 400 (UTS 400). The UTS 400 master station unit is shown in Figure 1–1.



44614

Figure 1–1. SPERRY UNIVAC Universal Terminal System 400 Master Station

General information concerning the UTS 400 is given in the UTS 400 system description, UP-8357 (current version), and operator information is given in the UTS 400 operator's guide, UP-8358 (current version). Program-language information is given in the current versions of the UTS 400 MAC 80 assembler programmer reference, UP-8482, and the UTS 400 utility library programmer reference, UP-8483. It is not within the scope of this manual to provide operating and servicing information.

1.2. UTS 400 OVERVIEW

The UTS 400 is a general-purpose, microprocessor-based remote display terminal system used for interactive* data communications with a central processing system. Communications can be conducted over any suitable communications network with any central processing installation that complies with the electrical and procedural requirements of the UTS 400.

Messages are composed and edited by the operator, using the keyboard, and are displayed on the screen before being transmitted to the host processor for data processing or transferred to a peripheral device for printing or storage. Messages from the host processor to the UTS 400 can also be displayed on the screen or routed to a peripheral.

The terminal system is available in two different configurations: as a master station with zero, one, or two slave stations, and as a controller unit with a minimum of one and up to a maximum of six slave stations. These two configurations are illustrated in Figures 1–2 and 1–3, respectively.

The master and slave stations have a keyboard and a cathode-ray-tube screen for use in message composition and display. The controller unit has no keyboard and no display screen. In controller configurations, messages are composed, edited, and displayed at the slave stations.

→ Optional SPERRY UNIVAC printers — the Model 800 Terminal Printer, the Communications Output Printer, the 0786 Printer Subsystem, or the 0791 Correspondence Quality Printer Subsystem — may be connected to the UTS 400 master station or controller to provide hard-copy printouts. An optional SPERRY UNIVAC Tape Cassette System or Diskette Subsystem may be connected to the UTS 400 master station or controller to provide local high-volume storage and retrieval. The printers and storage devices are referred to as share-type peripherals because the master station or controller and its possible slave stations share these peripherals.

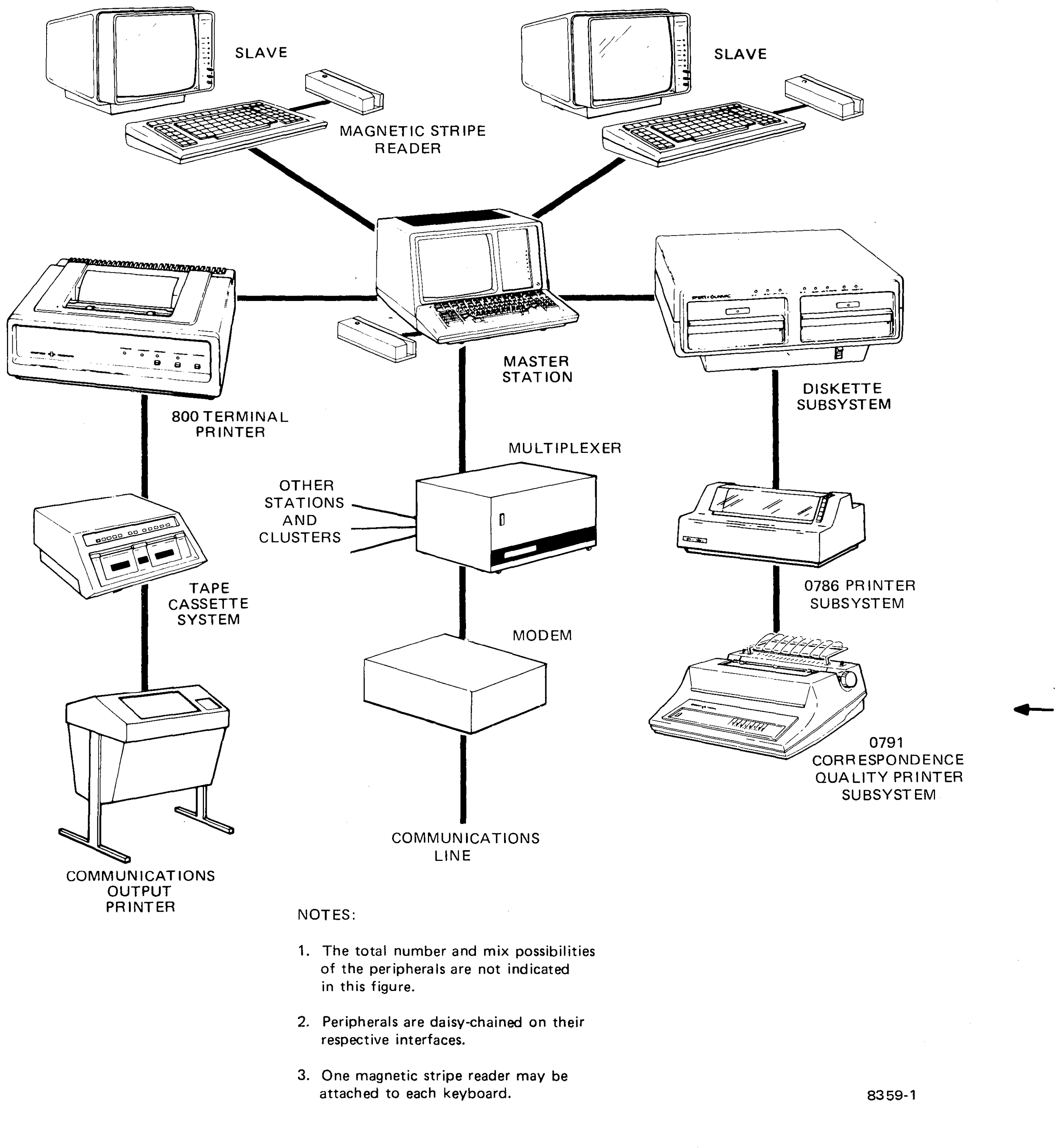
Another optional peripheral device, the SPERRY UNIVAC Magnetic Stripe Reader, reads data from the magnetic stripe on credit-card-type media and enters that data into the UTS 400. This read-only device may be connected to the UTS 400 through the keyboards of individual master or slave stations. The reader is referred to as a nonshare-type peripheral because it is not shared with the system as the other peripheral devices are.

1.3. UTS 400 COMMUNICATIONS ENVIRONMENT

1.3.1. System Configurations

The UTS 400 can communicate with the host processor over the public telephone network, on leased common-carrier voice-grade lines, or directly over a private communications line. As shown in Figure 1–4, the UTS 400 can be used in point-to-point or multidrop configurations. In a multidrop configuration, multiple UTS 400

**This term and others peculiar to data communications are defined in the glossary.*



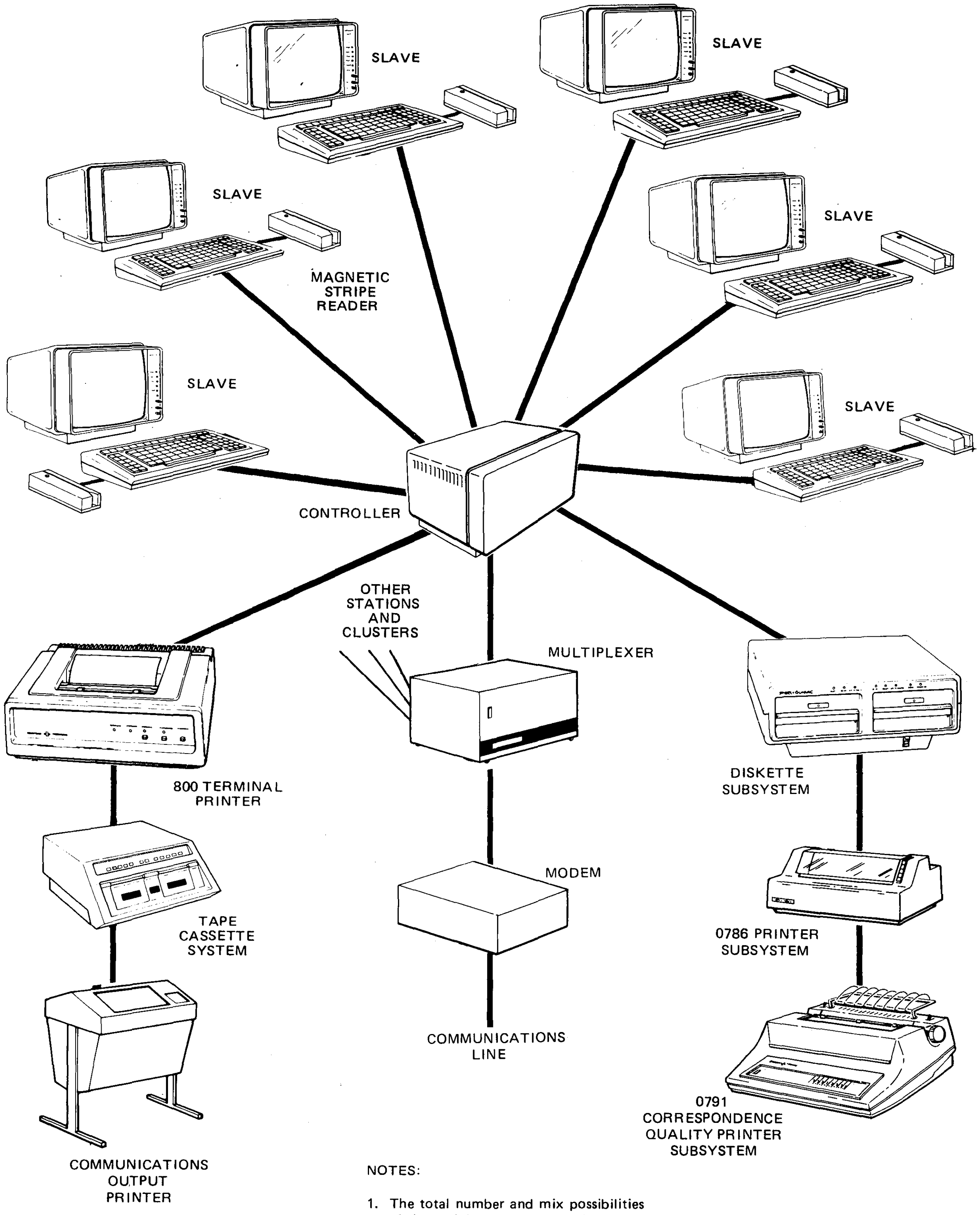
NOTES:

1. The total number and mix possibilities of the peripherals are not indicated in this figure.

2. Peripherals are daisy-chained on their respective interfaces.

3. One magnetic stripe reader may be attached to each keyboard.

Figure 1–2. UTS 400 Master Station With Slave Stations and Optional Peripheral Devices



NOTES:

1. The total number and mix possibilities of the peripherals are not indicated in this figure.

2. Peripherals are daisy-chained on their respective interfaces.

3. One magnetic stripe reader may be attached to each keyboard.

8359-2

Figure 1–3. UTS 400 Controller With Slave Stations and Optional Peripheral Devices

systems can be connected to a communications line at a single interface point by means of a multiplexer. The actual number of UTS 400 systems that can be accommodated at one system interface point depends on several factors, such as the expected amount of traffic and length of messages from individual terminal systems and the software handler techniques employed.

The UTS 400 can also be intermixed with other SPERRY UNIVAC terminals: the DCT 1000 Data Communications Terminal and the UNISCOPE 100 and 200 Display Terminals. A typical communications system configuration using the UTS 400 with UNISCOPE terminals is shown in Figure 1–5.

1.3.1.1. Modems

Modems provide the connection between the UTS 400 and the communications line and between the host processor and the communications line, as shown in Figure 1–5. These devices convert digital data to signals that can be transmitted over the communications line and vice versa.

1.3.1.2. Terminal Multiplexer

By means of the SPERRY UNIVAC Terminal Multiplexer, multiple UTS 400 masters, master/slave clusters, and controller/slave clusters can be connected to a communications system at one interface point, as shown in Figure 1–5. The multiplexer provides system connection for up to 16 UTS 400 systems (masters with or without slaves, controllers with slaves). The UTS 400 can be used on the same terminal multiplexer with both UNISCOPE 100 and 200 terminals. However, if cascading is employed on a multiplexer servicing UNISCOPE 100 or 200 terminals, the UTS 400 can be connected only to the primary multiplexer; connection to the cascaded multiplexer is not allowed.

The multiplexer permits synchronous or asynchronous full-duplex operation through common-carrier modems, or full-duplex synchronous operation directly with a central processor equipped with suitable communications line interfaces.

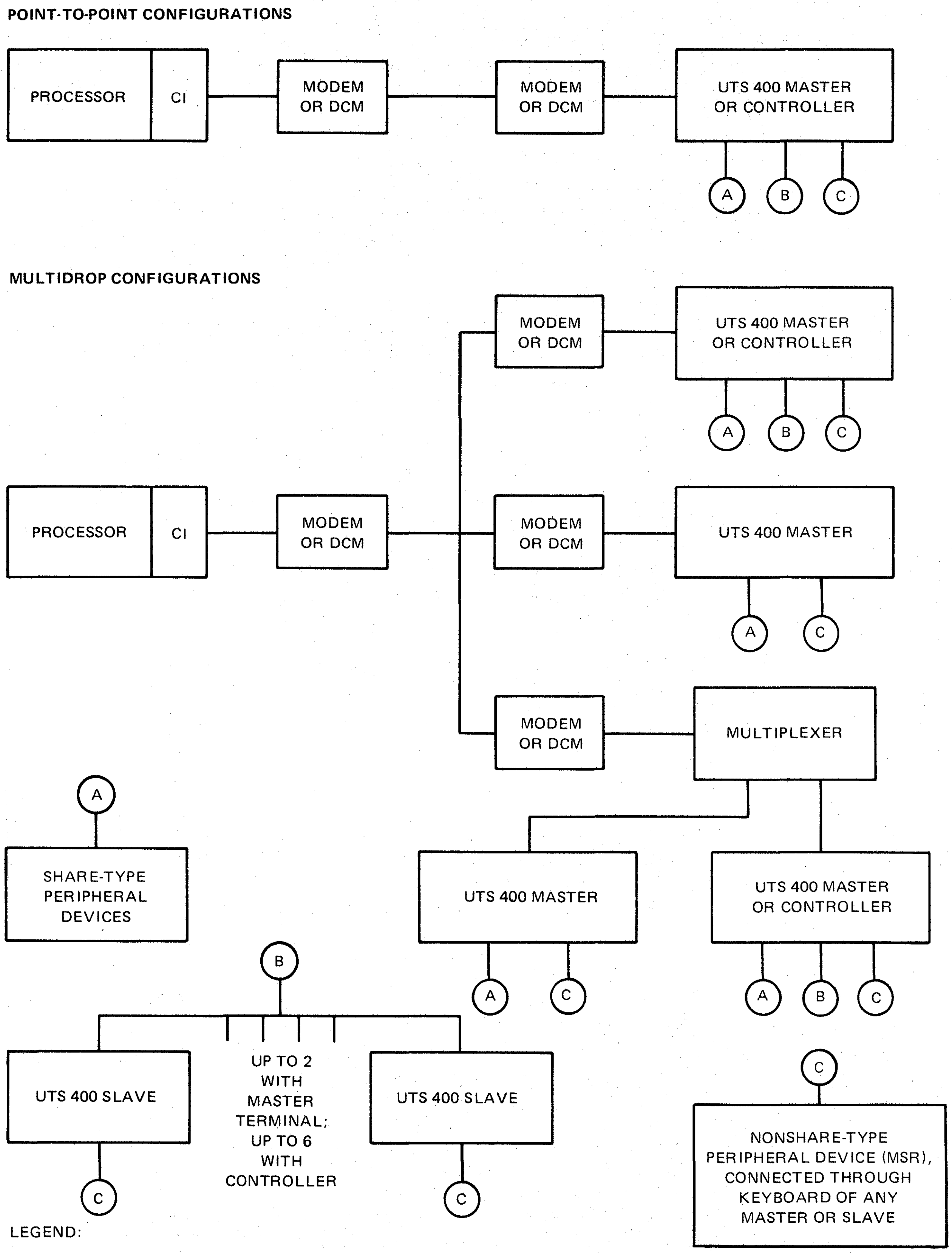
1.3.2. Transmission Characteristics

Transmission between the UTS 400 and the host processor is bit-serial. The data transmission code used is standard 7-level ASCII (American Standard Code for Information Interchange) plus character parity. The UTS 400 can transmit synchronously at speeds up to 9600 bits per second or asynchronously at 2400 bits per second. The UTS 400 operates in half-duplex mode; however, a full-duplex communications line can be used.

1.3.3. Message Routing

Messages from the processor are routed to a UTS 400 by means of message addressing. The messages are placed in the terminal system storage and are displayed on the screen. The optional printers, tape cassette system, and diskette subsystem can be accessed through the UTS 400 by way of the peripheral interface for printouts or for storage and retrieval of information.

An optional screen bypass feature is available which allows messages to be sent from the processor to a share-type peripheral or from a share-type peripheral to the processor without displaying the messages on the screen. ←

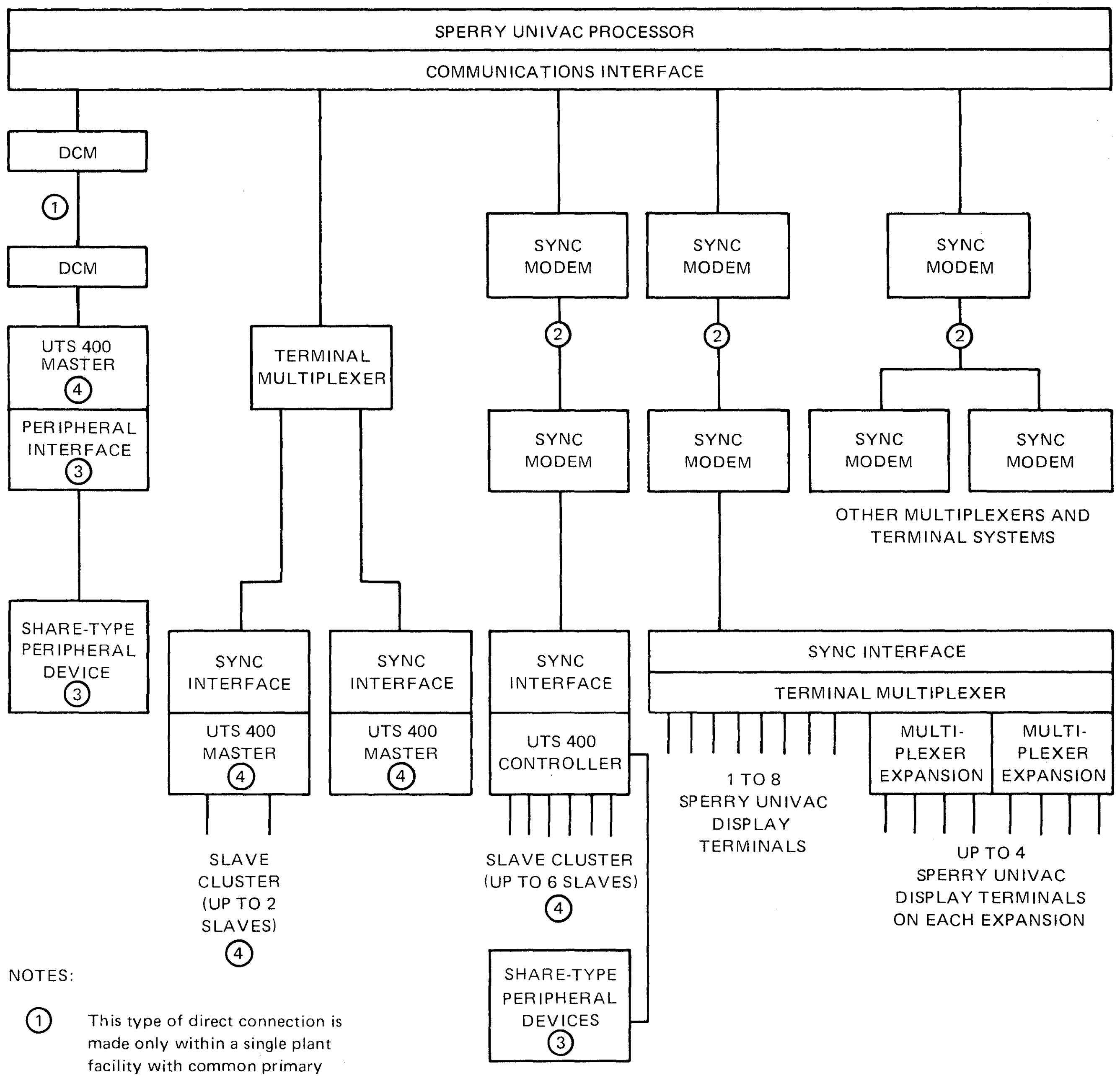

POINT-TO-POINT CONFIGURATIONS
PROCESSOR
CI
MODEM OR DCM
MODEM OR DCM
UTS 400 MASTER OR CONTROLLER
A
B
C
MULTIDROP CONFIGURATIONS
MODEM OR DCM
UTS 400 MASTER OR CONTROLLER
A
B
C
PROCESSOR
CI
MODEM OR DCM
MODEM OR DCM
UTS 400 MASTER
A
C
MODEM OR DCM
MULTIPLEXER
A
SHARE-TYPE PERIPHERAL DEVICES
UTS 400 MASTER
A
C
UTS 400 MASTER OR CONTROLLER
A
B
C
B
UTS 400 SLAVE
UP TO 2 WITH MASTER TERMINAL; UP TO 6 WITH CONTROLLER
UTS 400 SLAVE
C
C
C
NONSHARE-TYPE PERIPHERAL DEVICE (MSR), CONNECTED THROUGH KEYBOARD OF ANY MASTER OR SLAVE

LEGEND:

CI = COMMUNICATIONS INTERFACE
DCM = SPERRY UNIVAC DIRECT CONNECTION MODULE
MSR = MAGNETIC STRIPE READER

44617

Figure 1–4. UTS 400 Terminal System Configurations



NOTES:

1. This type of direct connection is made only within a single plant facility with common primary power throughout.

2. Switched or private lines. Asynchronous interfaces also are available.

3. Any UTS 400 master or controller can have a peripheral interface and one or more share-type peripheral devices. Slaves will share these devices.

4. Any UTS 400 master or slave can have one nonshare-type peripheral device (magnetic stripe reader) connected through its keyboard.

44618

Figure 1–5. Typical Communications System Configuration Using UTS 400 Systems and UNISCOPE Display Terminals

2. Functional Description

2.1. GENERAL INFORMATION

The UTS 400 master and slave stations have a display screen and each can have a keyboard. For configurations that have a controller, the display and keyboard functions are performed by the slave stations. Each UTS 400 cluster (master with slaves or controller with slaves) shares a microprocessor, working buffer storage, and other circuitry including the communications interface and peripheral interfaces. The shared components are located either in the master station or in the controller.

The control page functions determine the type of transmission from the UTS 400 to the host processor and the type of data transfer to and from the share-type peripherals. These functions, described in greater detail in 2.4, are set by the operator.

The field control character (FCC) may be used to condition the UTS 400 so that routine information can be entered in any desired format. By use of the FCC, fields may be defined to highlight selected elements of data, to automatically reject the wrong type of data, or to prevent the entry or change of any data in a given field. The FCC may also be used to specify that only changed or variable data be transmitted to the host processor or sent to the peripherals. Both the host processor and the operator at the keyboard have access to these FCC functions. The FCC is discussed in more detail in Section 4.

2.2. DISPLAY SCREEN

Data received from the host processor, composed from the keyboard by the operator, or read from a nonshare- or share-type peripheral is displayed on the screen of the UTS 400. The characters are entered on the screen one at a time at the location marked by the cursor.

The cursor (◪) is a unique character that is constantly displayed on the screen (except for a few moments when data is being transmitted or is being transferred via the peripheral interface) to mark the position that will be occupied by the next character entered into storage (and on the screen). The cursor also marks the last character of text when data is transmitted to the processor or sent to a share-type peripheral.

2.3. KEYBOARD

UTS 400 messages 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, and the cursor is advanced one position. Detailed operating instructions are provided in the UTS 400 operator's guide, UP-8358.

The basic keyboard includes alphanumeric keys, a numeric pad, editing and cursor control keys, shift keys, peripheral interface function keys, transaction keys, and a full set of program attention keys, as shown in Figure 2–1.

Either a 64-character, 96-character, or Katakana/English character set, and corresponding keyboard, may be used with the UTS 400. Keycap selection corresponds to the character set selected and the language required. The Katakana/English keyboard is a separate choice with no keycap selection required.

The domestic (USA) character set and the corresponding binary, octal, and hexadecimal codes are shown in Figure 2–2. The Katakana characters that can be generated by the Katakana/English character set are shown in Figure 2–3. The corresponding binary, octal, and hexadecimal codes for these Katakana characters are also shown in this figure. The foreign language characters that differ from the domestic character set are listed in Figure 2–4 and identified by the column and row where they fit in the UTS 400 code chart (Figure 2–2).

2.3.1. Program Attention Keys

The keyboard has 22 program attention keys, labeled F1 through F22. Each of these keys is a message generator that can be used by the operator to initiate special sequences or functions as designated in the host program. (The MESSAGE WAITING key, which functions in the same manner as the program attention keys, can be used to request a message generated or forwarded by the host processor.)

2.3.2. Transaction Keys

The transaction keys are XMIT (transmit), XFER (transfer), and PRINT. They are used by the UTS 400 to control activity to and from the host processor and the share-type peripherals. The functions of these keys are discussed in connection with the control page in 2.4.

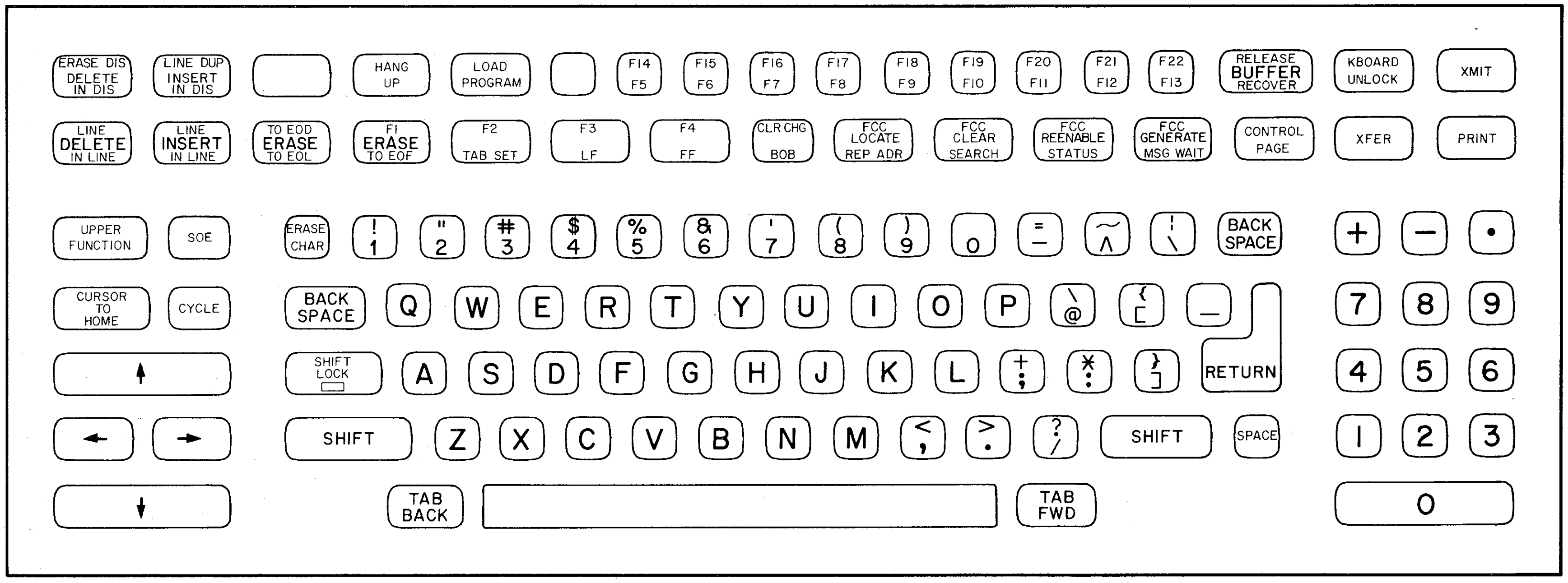


Figure 2–1. UTS 400 Keyboard

CONTROL CHARACTERS		DATA CHARACTERS						HEXADECIMAL				
		64 CHARACTERS (UPPERCASE)				32 CHARACTERS (LOWERCASE)						
0	1	2	3	4	5	6	7	← LEFT DIGIT				RIGHT DIGIT
0	0	0	0	1	1	1	1	7	BINARY BIT NUMBER			
0	0	1	1	0	0	1	1	6				
0	1	0	1	0	1	0	1	5 / 4	3	2	1	
NUL 000	DLE 020	SP 040	0 060	@ 100	P 120	` 140	p 160	0	0	0	0	0
SOH 001	DC1 021	! 041	1 061	A 101	Q 121	a 141	q 161	0	0	0	1	1
STX 002	DC2 022	" 042	2 062	B 102	R 122	b 142	r 162	0	0	1	0	2
ETX 003	DC3 023	# 043	3 063	C 103	S 123	c 143	s 163	0	0	1	1	3
EOT 004	DC4 024	$ 044	4 064	D 104	T 124	d 144	t 164	0	1	0	0	4
ENQ 005	NAK 025	% 045	5 065	E 105	U 125	e 145	u 165	0	1	0	1	5
ACK 006	SYN 026	& 046	6 066	F 106	V 126	f 146	v 166	0	1	1	0	6
BEL 007	ETB 027	' 047	7 067	G 107	W 127	g 147	w 167	0	1	1	1	7
BS 010	CAN 030	(050	8 070	H 110	X 130	h 150	x 170	1	0	0	0	8
HT 011	EM 031	) 051	9 071	I 111	Y 131	i 151	y 171	1	0	0	1	9
LF 012	SUB 032	* 052	: 072	J 112	Z 132	j 152	z 172	1	0	1	0	A
VT 013	ESC 033	+ 053	; 073	K 113	[133	k 153	{ 173	1	0	1	1	B
FF 014	FS 034	, 054	< 074	L 114	\ 134	l 154	¦ 174	1	1	0	0	C
CR 015	GS 035	- 055	= 075	M 115	] 135	m 155	} 175	1	1	0	1	D
SO 016	RS 036	. 056	> 076	N 116	∧ 136	n 156	~ 176	1	1	1	0	E
SI 017	US 037	/ 057	? 077	O 117	_ 137	o 157	⫽ 177	1	1	1	1	F
		RID			SID		DID				LS	

NOTES:

1. Octal codes appear in the square with each character.

2. General identifier (GID) characters are circled.

3. MS = most significant, LS = least significant.

4. The ⫽ symbol (column 7, octal code 177) is the UTS 400 symbol for the ASCII delete character (DEL).

5. Examples:

   A. For the dollar sign character ($), the octal code is 044, the hexadecimal code is 24, and the binary code is 0 1 0 0 1 0 0.
   7 6 5 4 3 2 1 ← BIT NUMBER

   B. Hexadecimal code 6E is for the lowercase (n) character.
   left digit → ← right digit
   The octal code is 156, and the binary code is 1 1 0 1 1 1 0.
   MS → ← LS

44620

Figure 2–2. UTS 400 Code Chart, Based on ASCII

			DATA CHARACTERS						HEXADECIMAL				
			64 CHARACTERS (UPPERCASE)										
MS	0	1	2	3	4	5	6	7	← LEFT DIGIT				RIGHT DIGIT
	0	0	0	0	1	1	1	1	7	BINARY			
	0	0	1	1	0	0	1	1	6	BIT NUMBER			
	0	1	0	1	0	1	0	1	5/4	3	2	1	↓
			SP Δ	ー	タ	ミ			0	0	0	0	0
			。	ア	チ	ム			0	0	0	1	1
			｢	イ	ツ	メ			0	0	1	0	2
			｣	ウ	テ	モ			0	0	1	1	3
			、	エ	ト	ヤ			0	1	0	0	4
			・	オ	ナ	ユ			0	1	0	1	5
			ヲ	カ	ニ	ヨ			0	1	1	0	6
			ァ	キ	ヌ	ラ			0	1	1	1	7
			ィ	ク	ネ	リ			1	0	0	0	8
			ゥ	ケ	ノ	ル			1	0	0	1	9
			ェ	コ	ハ	レ			1	0	1	0	A
			ォ	サ	ヒ	ロ			1	0	1	1	B
			ャ	シ	フ	ワ			1	1	0	0	C
			ュ	ス	ヘ	ン			1	1	0	1	D
			ョ	セ	ホ	゛			1	1	1	0	E
			ッ	ソ	マ	゜			1	1	1	1	F

LS

NOTES:

1. In this chart, only the Katakana characters generated by the Katakana/English character set are shown. Binary, octal, and hexadecimal codes are represented the same way as in Figure 2–2.

2. Control characters and English characters generated by the Katakana/English character set and their corresponding codes are the same as those shown in Figure 2–2.

44621

Figure 2–3. UTS 400 Katakana/English Code Chart, Based on ASCII

COLUMN / ROW	2/3	2/4	4/0	5/11	5/12	5/13	5/14	6/0	7/11	7/12	7/13	7/14
DOMESTIC (USA)	#	$	@	[	\	]	^	ˋ	{	¦	}	~
SPAIN	Pt	$	@	¡	Ñ	¿	^	ˋ	°	ñ	£	~
DENMARK / NORWAY	#	$	@	Æ	Ø	Å	^	°	æ	ø	å	—
FRANCE	£	$	à	°	ç	§	^	ˋ	é	ù	è	ê
FRENCH AZERTY	#	$	à	°	ç	§	^	ˋ	é	ù	è	ê
GERMANY	#	$	§	Ä	Ö	Ü	^	ˋ	ä	ö	ü	ß
SWEDEN	#	$	É	Ä	Ö	Å	Ü	é	ä	ö	å	ü
UNITED KINGDOM	£	$	@	[	\	]	↑	ˋ	{	│	}	—

44622

Figure 2–4. UTS 400 Foreign-Language Character Locations on Code Chart

2.4. CONTROL PAGE

The control page is an array of control and status information stored in a dedicated location of the UTS 400 memory. This 2-line display is used by the operator in conjunction with several of the peripheral interface function keys to control the transfer of data to and from the share-type peripherals and to control the type of transmission from the UTS 400 to the host processor.

The operator calls the control page display to the screen with the CONTROL PAGE key. When the control page is called to the screen, any data present on the first two display lines is shifted to temporary storage in memory. This data is returned to the display when the control page is removed from the screen. The control page format is:

```
(**PRINT*)STA-    (**XFER**)PRNT(    )XFER(    )XMIT(    )MM
( / / )ADR-       ( / / )SEARCH(                         )laD
```

NOTE:

The characters occupying the last three positions in the second line of the control page define the remote identifier (RID), the station identifier (SID), and the firmware revision level for the station. (The three characters shown here are examples only.) The SID range is octal codes 120 through 157, whose values are represented by the lowercase letters a through o. If the UTS 400 has a 64-character display rather than a 96-character display, all lowercase letters are displayed as uppercase letters (A through O).

The operator specifies in the control page the share-type peripherals to be selected. Thereafter, these peripherals are selected by the UTS 400 each time a transfer, print, backward one block, report address, search, or status key is pressed. When a different peripheral is desired for a particular function, the appropriate control page field must be redefined.

The transmit, transfer, and print functions are initiated by pressing the XMIT, XFER, and PRINT keys, respectively. The key pressed initiates the function as defined in the corresponding field of the control page. When a different function is desired for one of these keys, the appropriate control page field must be redefined.

The transmit function causes data to be sent to the host processor by way of the communications line; the transfer and print functions cause data to be moved to and from the share-type peripherals.

2.5. CONTROLS AND INDICATORS

UTS 400 controls and indicators used in basic operation of the terminal system are listed in Table 2–1. Those marked with an asterisk (*) are located on the slave station, and those marked with a dagger (†) are located on the controller. All of the controls and indicators listed are located on the master station.

Table 2–1. UTS 400 Controls and Indicators (Part 1 of 2)

Designation	Control or Indicator	Function
POWER*†	Control	Applies or removes primary power.
	Indicator	Lights when power is applied to the UTS 400.
WAIT*	Indicator	Lights when a text message is being transmitted to or from the UTS 400. At the same time, indicates that the keyboard is locked (functionally disabled). Also lights during a peripheral interface data transfer to or from the screen.
INTENSITY*	Control	Adjusts brightness of the screen display.
MESSAGE WAITING*	Indicator	Lights when the processor has a conditional unsolicited message for display. Stays on until the MSG WAIT key or a special function key is pressed and the processor-message-waiting request or function-key message is sent.
MESSAGE INCOMPLETE*	Indicator	Lights during the time a text message is being received by the UTS 400. Goes out when all checks for the message have been satisfied.
Audible alarm*	Indicator (tone)	Sounds once when the cursor moves into the eighth character position from the right on any line in the display. Sounds once when the cursor first moves into the last line (in any character position) and again when it reaches the eighth character position from the end of the last line of the display. Sounds intermittently during the time that the MESSAGE WAITING indicator is lit. The alarm is turned off when the MESSAGE WAITING indicator is turned off. Sounds when an error is made in manually entering a field control character. Sounds when the peripheral interface automatic recovery fails. When the UTS 400 is operating in the character-protection mode, sounds each time an attempt is made to enter a data character at the cursor position when the character under the cursor is a UNISCOPE terminal protected character.
TEST†	Control	Initiates a test involving a diagnostic program internal to the UTS 400. The test leaves the terminal system in a predetermined condition.

Table 2–1. UTS 400 Controls and Indicators (Part 2 of 2)

Designation	Control or Indicator	Function
MONITOR†	Control	Causes the information on the communications line to be displayed on the screen.
READY*†	Indicator	Lights if the unit is functioning properly.
POLL*†	Indicator	Blinks when the unit is being polled.
FCC/PROTECT†	Control	Allows the selection of protected or FCC field definition on communications transactions. NOTES: The PROTECT position cannot be used with the Katakana capability. When the UTS 400 is equipped with the optional character-protection feature, this switch must be in the FCC position for the optional character-protection feature to function.
AUXILIARY BUSY*	Indicator	Lights when the peripheral interface is busy and unable to accept further data.
SHIFT LOCK*	Indicator	Lights when the keyboard is in the uppercase condition. This indicator is located in the SHIFT LOCK key. (On Katakana/English keyboards, the KANA SHIFT key also has a shift indicator.)

2.6. UTS 400 INTERNAL OPERATION

The UTS 400 is an "intelligent" terminal system. The intelligence is provided by a microprocessor which makes logical decisions based upon firmware program instructions and electrical conditions within the UTS 400.

The firmware functions, housed in read-only memory (ROM), are always available to the UTS 400. Firmware does not have to be loaded before the terminal system will function, nor is it destroyed in the event of a power failure. The firmware operating system is an interrupt-driven, serial-reentrant program. The firmware controls the UTS 400 by moving data from one interface point to another in response to control codes initiated from the host processor, the keyboard, or the peripherals. The firmware also maintains the display information for each station and routes the data to and from the display memories.

The following functions are effected by a combination of hardware and firmware:

- The communications interface, which allows the UTS 400 to be connected to a host processor by way of a communications line. All communications line protocol, including polling responses, data handling, and error recovery notifications, is incorporated in this interface.

- Management of the display screen information and of the keyboard or magnetic stripe reader nonshare-type peripheral input.

- The peripheral interfaces, which provide the control for all data movement to and from the share-type peripherals. These interfaces allow all peripherals to be shared by all stations in a UTS 400 cluster; however, only one peripheral can be active at a given time.

The interaction of these control functions is shown in the data flow diagram in Figure 2–5.

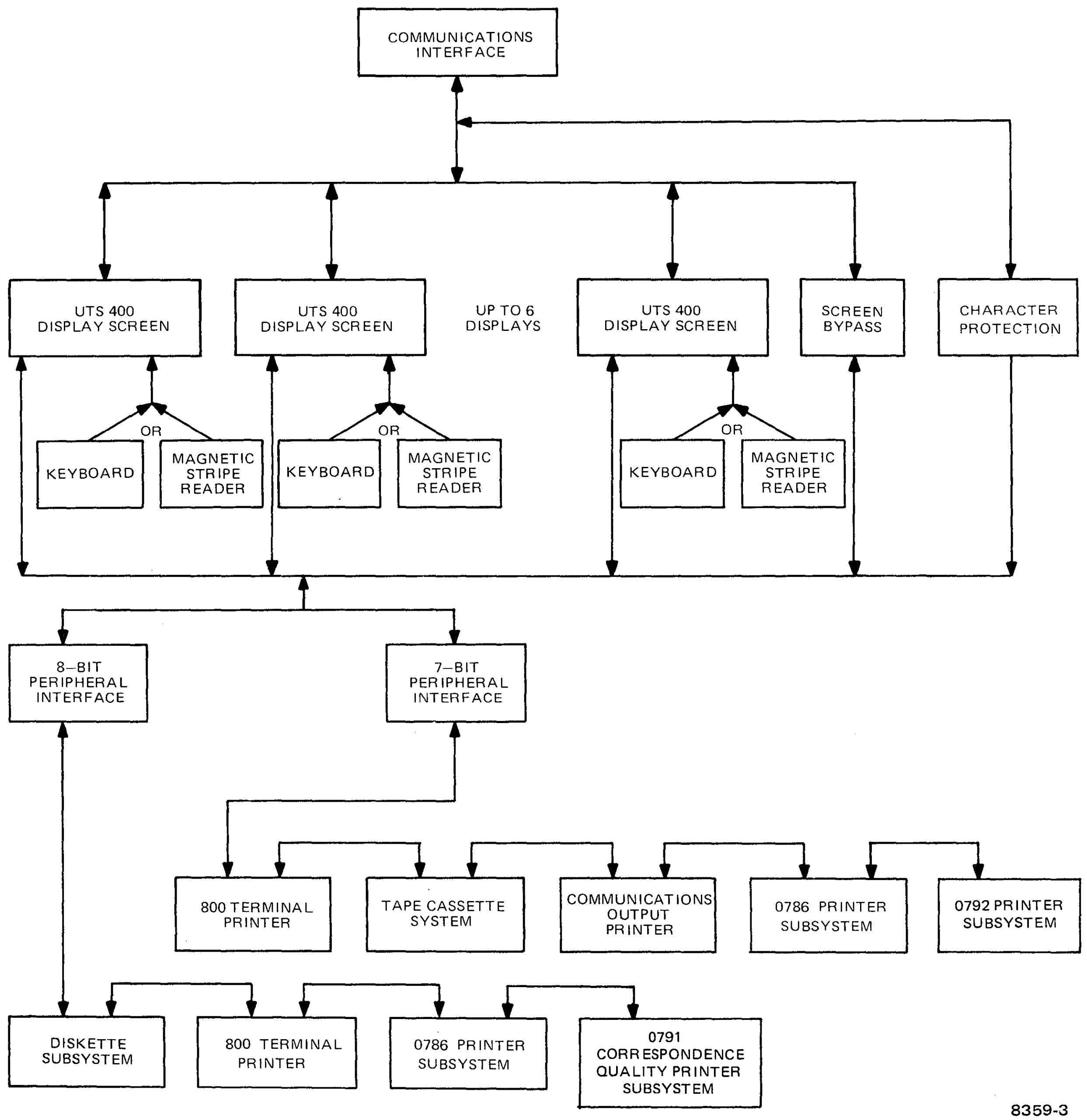


Figure 2–5. UTS 400 Data Flow Diagram

3. Communications Protocol

3.1. GENERAL INFORMATION

The UTS 400 provides interactive communication with a host processor over a communications line. Interchange between the UTS 400 and the host processor consists of a sequence of messages, and this message interchange is governed by a series of rules that allow information to be conveyed in either direction in a standard manner.

Throughout this manual, message-exchange sequences are represented as follows:

Host Processor	**Direction of Transmission**	**UTS 400**
Messages from the host processor are listed on the left with an arrow pointing to the right (to the UTS 400).	$\longrightarrow$	
	$\longleftarrow$	Messages from the UTS 400 are listed on the right with an arrow pointing to the left (to the host processor).

In the message sequences, acronyms representing required characters appear in all capital letters, and terms indicating variable information appear in lowercase letters.

NOTE:

The rules for host processor and UTS 400 communication interchange that are detailed in this section are summarized in Appendix A. Communications control sequences are summarized in Appendix B.

3.1.1. Poll/Response Message Sequence

The basic element in the UTS 400 communications protocol is an error-free poll from the host processor. A poll is a message from the host processor that requests a response from the UTS 400. The UTS 400 can send a message to the host processor only when polled by the host processor.

A simplified basic example of a complete communications message sequence between the UTS 400 and the host processor is as follows:

Host Processor	**Direction of Transmission**	**UTS 400**
Poll	⟶	
	⟵	Acknowledgeable response
Poll with acknowledgment	⟶	
	⟵	No traffic

In the preceding example, the poll is soliciting traffic from the UTS 400. The UTS 400 station responds with a message conveying data, status, or a combination thereof. The host processor signifies proper receipt of that message from the station by including an acknowledgment in the next poll to that station. The station responds with a no-traffic message, indicating that it received the acknowledgment and has nothing further to send at this time.

3.1.2. Host Processor Text to UTS 400 – Message Sequence

After sending text to a UTS 400 station, the host processor must always send a poll to verify that the station received the text message correctly. If the poll response from the station contains an acknowledgment, then the text was received correctly by the station. If the poll response does not contain an acknowledgment, then the text was not received correctly by the station.

The basic message sequence required to send text from the host processor to a UTS 400 station is as follows:

Host Processor	**Direction of Transmission**	**UTS 400**
Text	⟶	
Poll	⟶	
	⟵	Response containing an acknowledgment*
Poll with acknowledgment	⟶	
	⟵	No traffic

In the preceding message sequence, the host processor sends text to the UTS 400 station and then polls to determine whether or not the text was received correctly (that is, with good character and block parity). The station responds to the poll with an acknowledgment, indicating it received the text correctly. The host processor then signifies proper receipt of the station acknowledgment by including an acknowledgment in the next poll to that station. The station then responds with a no-traffic message, indicating that it received the acknowledgment and has nothing further to send.

**The two possible forms of acknowledgment from the UTS 400 station are defined in 3.4.2.3.*

3.2. STATION/PERIPHERAL ADDRESSING

NOTE:

The station/peripheral addressing scheme discussed in the following paragraphs refers to the share-type peripheral devices that require a peripheral interface. Since the magnetic stripe reader is a nonshare-type peripheral device, it is not to be considered as part of this discussion.

Stations on a communications line are distinguished from one another by an addressing scheme that identifies the intended recipient(s) of a host processor message and, in like manner, identifies the responding station.

The UTS 400 addressing scheme utilizes a 3-character sequence consisting of a remote identifier (RID), a station identifier (SID), and a peripheral device identifier (DID). The possible RID, SID, and DID characters are shown in Figure 2–2.

Of the possible 48 RIDs, 47 are specific (assigned to specific stations) and 1, the SP character, is general (recognized by all stations). Of the 32 SIDs, 31 are specific and 1, the P character, is general. Of the 16 DIDs, 15 are specific and 1, the p character, is general.

Each station on a communications line is uniquely identified by a specific RID and specific SID combination. Thus, the address of a station is symbolized by RID SID. Each peripheral accessed by a station is uniquely identified by a specific DID. Specific DIDs as used for peripheral selection are discussed in 3.8.3.

All messages (except no-traffic) contain the 3-character address, symbolized by RID SID DID, immediately following the start-of-header (SOH) character. (Refer to 3.4 for message formats.)

3.2.1. Hardware Setting of Addresses

The RID and SID of the UTS 400 master station or of the primary slave station in a UTS 400 controller/slave cluster are set by switches on the communication I/O board located in the master station or controller, respectively. (Since the controller itself is not a station, one of the slave stations is designated as the primary station by switch settings on the communication I/O board in the controller.) There are no switch settings in the slave stations to define their RID SID addresses. Additional slave stations associated with the master station or primary slave all have the same RID but have unique SIDs that are assigned, by firmware, in ascending, sequential order from the SID of the master or primary slave station. If the screen bypass feature is present, it also has the same RID, but its SID is the next in sequence above that of the last station present.

3.2.2. Poll Groups

A poll group is defined as all stations on the communications line that recognize the RID and SID of a poll message. Each station in such a group will recognize a general RID as its RID address and a general SID as its SID address. A poll containing a general RID, general SID, and general DID is called a general poll. Another form of general poll is one containing a specific RID, general SID, and general DID.

NOTE:

The general polls used in the examples in this section all contain a specific RID, general SID, and general DID.

A poll containing a specific RID, specific SID, and general DID is called a specific poll.

Figure 3–1 illustrates three examples of poll groups that would respond to a general poll (specific RID, general SID, general DID) from the host processor. The UTS 400 master/slave cluster defines a poll group with the common RID of 1. The UTS 400 controller/slave cluster defines a poll group with the common RID of 2. The three UTS 400 masters with a common RID of 3 define a third poll group. Each of these poll groups is connected on a multidrop line. Note that, in this figure, each poll group corresponds to a drop on the multidrop line.

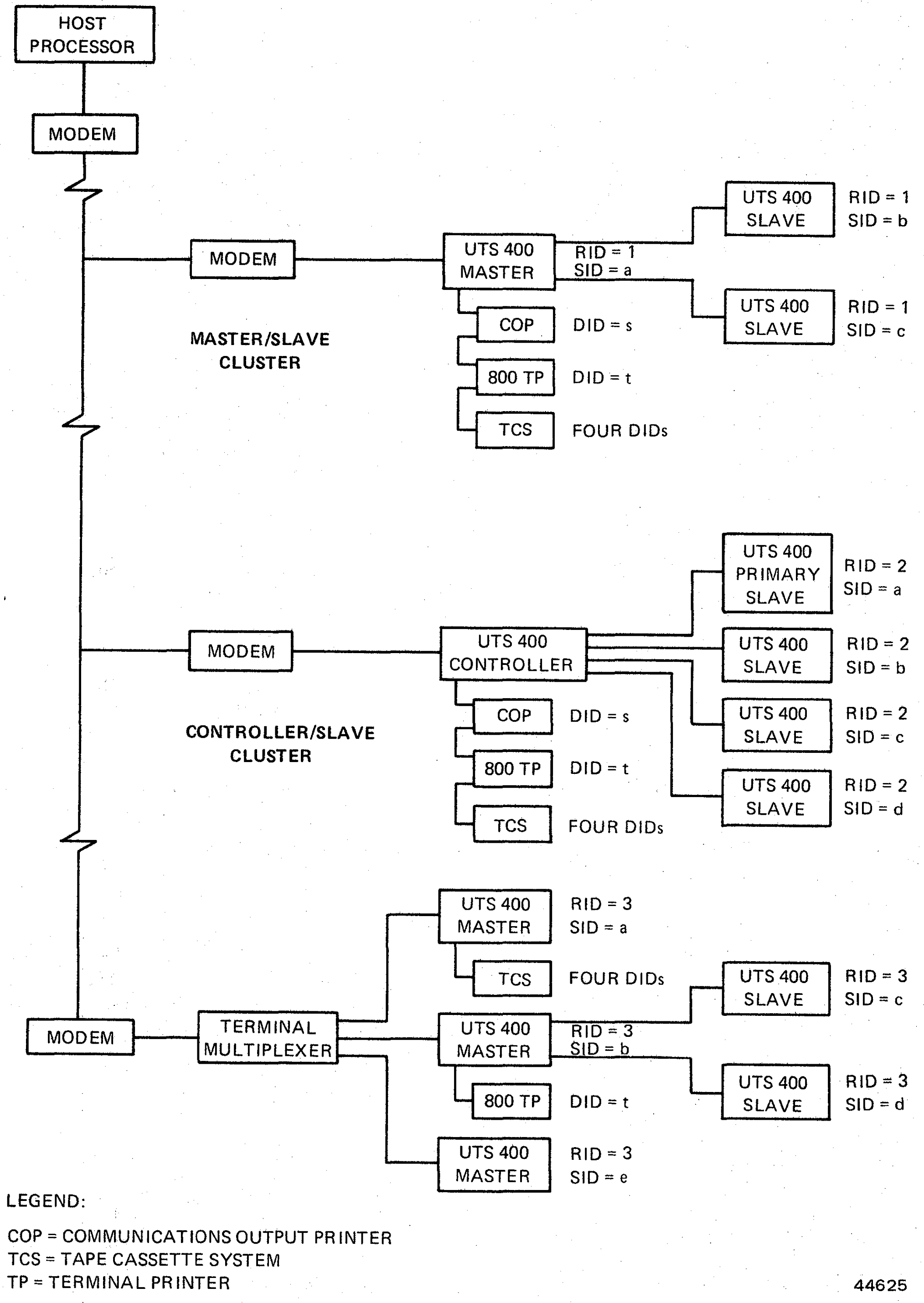


Figure 3–1. Example of UTS 400 Address Code Assignments

All the stations in a poll group addressed by a general poll are candidates to furnish the response. The UTS 400 master and the UTS 400 controller both perform a function similar to that of the terminal multiplexer (described in 3.6) in terms of determining which of the associated stations within the poll group will be allowed to respond.

No two drops on a multidrop line can be members of the same poll group, since both drops would try to respond at once because there is no means to resolve the contention for the line. Note, however, that more than one poll group may be associated with a terminal multiplexer (if multiple RIDs are assigned), hence providing multiple poll groups at a physical drop.

3.3. POLL MESSAGE TYPES

The host processor can send the following types of polls to the UTS 400 station:

- Status poll
- Traffic poll
- Selection poll
- Retransmission request

3.3.1. Status Poll

A status poll solicits any nontext message the station has to send. If the station has a text message to send, then the status poll response contains the DLE 0 character sequence, indicating that text is available. The status poll can be specific or general. A specific status poll requests status from a particular station, and a general status poll requests status from a poll group.

3.3.2. Traffic Poll

A traffic poll solicits any message the station has to send, including text messages. The traffic poll can be specific or general. A specific traffic poll requests traffic from a particular station, and a general traffic poll requests traffic from a poll group.

3.3.3. Selection Poll

The selection poll has a specific RID, a specific SID, and a specific DID. Its primary purpose is to select a peripheral and obtain peripheral status. Peripheral selection is discussed in 3.8.3.

3.3.4. Retransmission Request

The retransmission request has a specific RID, a specific SID, and a general DID. The retransmission request causes the station to resend its last response and thus plays an important role in recovery from line errors. Line error recovery is discussed in 3.7.

3.4. MESSAGE FORMAT

All messages begin with the SOH character, with the exception of the no-traffic poll response.

On synchronous lines, the SOH character must be preceded by a minimum of four SYN characters.

All messages end with the end-of-text (ETX) character and then the block check character (BCC). The BCC facilitates line error detection, as described in 3.4.3.

The following paragraphs describe the format for host processor and UTS 400 messages.

3.4.1. Host Processor Messages

The host processor sends two types of messages: polls and text.

3.4.1.1. Poll Messages

The format of a poll is:

SOH RID SID DID control characters ETX BCC

The control characters that can be used in a poll message are shown in Table 3–1.

Table 3–1. Poll Message Control Characters

Poll Function	ASCII Characters	Octal Value	Hexadecimal Value
Traffic poll without acknowledgment	None required	–	–
Selection poll without acknowledgment*	None required	–	–
Retransmission request	DLE NAK	020 025	10 15
Traffic poll with acknowledgment	DLE 1	020 061	10 31
Selection poll with acknowledgment*	DLE 1	020 061	10 31
Status poll without acknowledgment	ENQ	005	05
Status poll with acknowledgment	DLE 1 ENQ**	020 061 005	10 31 05

*Peripheral selections can be attempted with status polls as well as with selection polls.

**The codes in this sequence may also be sent in reverse order as ENQ DLE 1.

3.4.1.2. Text Messages

3.4.1.2.1. Normal Text

Format:

SOH RID SID DID STX data ETX BCC

The data portion of host processor text messages is discussed in Section 4. All text messages must be addressed with the specific RID and specific SID of the station that is to receive the text.

3.4.1.2.2. Message Wait Command

Format:

SOH RID SID DID BEL STX ETX BCC

The message wait command turns on the MESSAGE WAITING indicator and the audible alarm of the UTS 400 station identified by the RID and SID in the message.

3.4.1.2.3. Disconnection Command

Format:

SOH RID SID DID DLE EOT STX ETX BCC

The disconnection command is discussed in 3.10.

3.4.1.2.4. Unsolicited Messages

Unsolicited messages sent to the UTS 400 may or may not be accepted, depending on the status condition of the UTS 400 at the time the unsolicited message is received.

A dump transmit sequence allows the UTS 400 in a transmit condition to accept host processor text by clearing the transmit condition and overwriting the data to be transmitted. The dump transmit format is as follows:

$$\text{SOH RID SID DID STX } \underbrace{\overbrace{\text{NUL . . . NUL}}^{20\text{ ms}}\text{ HT }\overbrace{\text{NUL . . . NUL}}^{40\text{ ms}}\text{ STX}}_{\text{Dump transmit command}}\text{ data}$$

A dump print sequence allows the UTS 400 in a print condition to accept host processor text by clearing the print condition and overwriting the data to be printed. The dump print format is as follows:

$$\text{SOH RID SID DID }\overbrace{\text{NUL . . . NUL}}^{40\text{ ms}}\text{ STX (Dump transmit command may follow.)}$$

3.4.2. UTS 400 Messages

All UTS 400 messages are sent in response to a poll and can be categorized as follows:

- Reply request
- No traffic
- Acknowledgment
- Traffic

3.4.2.1. Reply Request

Format:

SOH RID SID DID DLE ENQ ETX BCC

The reply request is discussed in 3.7.

3.4.2.2. No Traffic

Format:

EOT EOT ETX BCC

→ If the UTS 400 has no other appropriate response to send, it will send a no-traffic response. (This is the only UTS 400 response that does not require an acknowledgment from the host processor.)

3.4.2.3. Acknowledgment

An acknowledgment from the UTS 400 will occur in one of two forms:

- The acknowledge (ACK) response, which means that the host processor text was received without error. The ACK format is:

  SOH RID SID DID DLE 1 ETX BCC

- The busy (wait before transmit, or WABT) response, which means that the host processor text message was received without error and it attempted a peripheral operation that had not been completed prior to the arrival of the poll. The busy (WABT) format is:

  SOH RID SID DID DLE ? ETX BCC

3.4.2.4. Traffic

The UTS 400 can respond to either general or specific polls with traffic. The traffic response may or may not carry an ACK or WABT with it. The traffic response can be any one, but only one, of the following conditions:

- Text data from a station screen when a transmit condition exists
- Program attention key codes when one of the UTS 400 program attention keys has been pressed
- Status of one of the share-type peripheral devices sent as a result of a selection attempt ◄—
- Status of a UTS 400 station
- Disconnection message from a UTS 400 station

3.4.2.4.1. Text as Traffic

Format:

SOH RID SID DID (DLE 1 or DLE ?)* STX data ETX BCC

The data portion of UTS 400 text messages is discussed in Section 4.

3.4.2.4.2. Program Attention Key Codes as Traffic

Format:

SOH RID SID DID (DLE 1 or DLE ?)* key code ETX BBC

The key code is a 1-character ASCII value. These messages are transmitted when the operator presses the MSG WAIT key or one of the program attention keys. The key codes are shown in Table 3–2.

3.4.2.4.3. Peripheral Status as Traffic

Format:

SOH RID SID DID (DLE 1 or DLE ?)* peripheral status ETX BCC

The peripheral status response consists of two characters, the first of which is always the DLE character. The possible peripheral status control sequences are shown in Table 3–3. (Peripheral status is discussed in 3.8.)

**ACK or WABT (DLE 1 or DLE ?) may or may not be included with the message.*

Table 3–2. Program Attention Key Codes

Key Label	ASCII Character	Octal Value	Hexadecimal Value
MSG WAIT	BEL	007	07
F1	7	067	37
F2	G	107	47
F3	W	127	57
F4	g	147	67
F5	Space	040	20
F6	!	041	21
F7	"	042	22
F8	#	043	23
F9	$	044	24
F10	%	045	25
F11	&	046	26
F12	'	047	27
F13	(	050	28
F14	)	051	29
F15	*	052	2A
F16	+	053	2B
F17	,	054	2C
F18	–	055	2D
F19	.	056	2E
F20	/	057	2F
F21	0	060	30
F22	1	061	31

Table 3–3. UTS 400 Peripheral Status Control Codes

Peripheral Status	ASCII Characters	Octal Value	Hexadecimal Value
Peripheral status 1 (ready)	DLE >	020 076	10 3E
Peripheral status 2*	DLE <	020 074	10 3C
Peripheral status 3*	DLE :	020 072	10 3A
Peripheral status 4 (no response)	DLE =	020 075	10 3D

*Status meaning depends on peripheral device

3.4.2.4.4. Station Status as Traffic

Format:

SOH RID SID DID (DLE 1 or DLE ?) station status ETX BCC**

The station status response consists of two characters, the first of which is always the DLE character. The possible station status control sequences are shown in Table 3–4. (The message-queued station status, the THRU station status, and the peripheral-selection-delayed station status are discussed in 3.8.)

Table 3–4. UTS 400 Station Status Control Codes

UTS 400 Status	ASCII Characters	Octal Value	Hexadecimal Value
Message queued	DLE 4	020 064	10 34
Peripheral selection delayed	DLE 5	020 065	10 35
Power-on confidence test completed	DLE 6	020 066	10 36
THRU (station is through with peripheral interface)	DLE ;	020 073	10 38
Text-available response to status poll	DLE 0	020 060	10 30

***ACK or WABT (DLE 1 or DLE ?) may or may not be included with the message*

3.4.2.4.5. Disconnection Message as Traffic

Format:

SOH RID SID DID DLE EOT ETX BCC

The disconnection message is discussed in 3.10.

3.4.3. Parity Checking

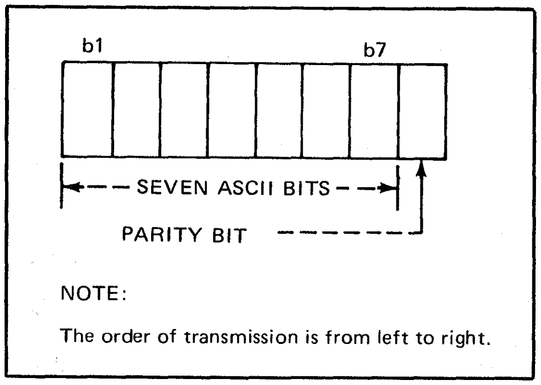
The UTS 400 uses two forms of parity checking – character and block parity – to insure data integrity on the communications line.

3.4.3.1. Character Parity

The character structure of data accepted by and sent from the UTS 400 conforms to American National Standard X3.16–1966.* Bit sequencing conforms to American National Standard X3.15–1966.** The character structure and bit sequencing standards applicable to the UTS 400 are summarized for synchronous and asynchronous transmission in the following paragraphs.

3.4.3.1.1. Synchronous Transmission

The character structure for synchronous data communications consists of eight bits: seven ASCII character bits plus one character parity bit (Figure 3–2).



44626

Figure 3–2. Character Structure, Synchronous Transmission

**Character Structure and Character Parity Sense for Serial-by-Bit Data Communications in the USA Standard Code for Information Interchange*

***Bit Sequencing of the USA Standard Code for Information Interchange in Serial-by-Bit Data Transmission*

The bit sequence for an ASCII character is from the least significant bit (b1) to the most significant bit (b7), in terms of the ASCII nomenclature,* in ascending order.

When transmitting, the UTS 400 generates a parity bit and adds it to every 7-bit code transmitted. When receiving, the UTS 400 checks the character parity. The character parity for synchronous transmission is odd, that is, an odd number of 1 bits per character.

3.4.3.1.2. Asynchronous Transmission

The character structure for asynchronous data communications consists of 10 signal elements of 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 3–3.)

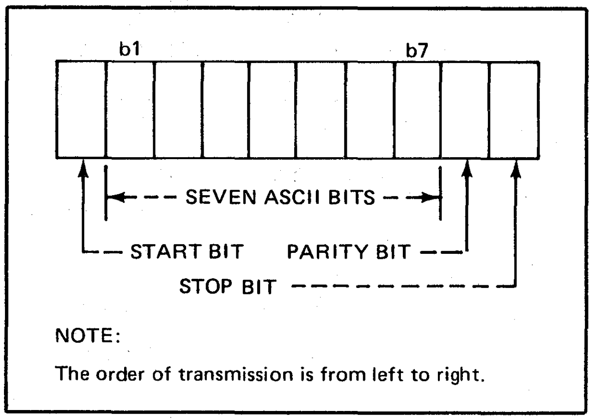


Figure 3–3. Character Structure, Asynchronous Transmission

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

When transmitting, the UTS 400 adds the start, parity, and stop bits to the seven ASCII bits. When receiving, the terminal establishes bit and character timing, using the start and stop bits; 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.

American National Standard Code for Information Interchange, X3.4–1966

3.4.3.2. Block Parity

A block check character (BCC) is accumulated by the UTS 400 on host processor messages and compared with the BCC supplied within the message to assure message validity. In like manner, the UTS 400 accumulates and includes a BCC on messages sent, thus allowing host processor validation of messages.

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 3–5.

Table 3–5. Example of BCC Parity Character Coding

Message	b7	b6	b5	b4	b3	b2	b1	Character Parity	
								Sync	Async
SOH	0	0	0	0	0	0	1	0	1
5	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.
3. The SYN character may be used as a time-fill character for synchronous transmission and must be excluded as part of the BCC calculation.

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

The BCC accumulation starts with (but does not include) the SOH character and ends with (and does include) the ETX character. All intervening characters (except SYN) are included in the accumulation.

The no-traffic response is a special case, since there is no SOH. For that response, the BCC accumulation corresponds to the ETX character; that is, the format is as follows:

EOT EOT ETX ETX

In either case, the BCC must immediately follow the ETX character. SYN characters must not be transmitted between ETX and BCC.

3.4.4. Communications Timing Considerations

Certain UTS 400 functions involve sequential access to all or part of the terminal storage. The execution times required to perform these functions are shown in Table 3–6.

Once one of these functions has been initiated, the terminal storage cannot be accessed until the function is completed. Characters from the communications line or share-type peripherals, except NULs, are put into a save area. This poses no problem when the function is initiated from the keyboard, since successive key depressions cannot occur rapidly enough to allow interference between functions. However, when any of the functions listed in Table 3–6 are initiated by way of the communications interface, there is a possibility that the limits of the save area may be exceeded. Therefore, it is suggested that each function be followed with enough NUL characters, at the transmission speed used, to keep the save area from overflowing.

The execution times listed in Table 3–6 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.

None of these functions require time fill characters at all times. If the time required to complete a function is less than the time fill value of the NUL character, no time fill is required. For example, erasing the 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 is required for this operation.

Table 3–6. Execution Times for UTS 400 Functions

Function and Control Code Sequence	Execution Time per Character Involved in the Function*
Backward tab (ESC z)	30 μs
Forward tab (HT)	30 μs
Delete in display (ESC C)	40 μs
Delete in line (ESC c)	40 μs
Delete line (ESC k)	30 μs
Erase unprotected data (ESC a)	20 μs
Erase display (ESC M)	10 μs
Erase to end of field (ESC K)	10 μs
Erase to end of line (ESC b)	10 μs
Insert in display (ESC D)	40 μs
Insert in line (ESC d)	40 μs
Insert line (ESC j)	30 μs
Line duplication (ESC y)	5 ms**
Control page access (ESC o)	5 ms**
Dump transmit	40 ms**
Dump print	40 ms**

*Time required is dependent upon application and can vary from zero time fill up to the maximum shown in the table.

**Time for the complete function.

3.5. BASIC COMMUNICATIONS PROTOCOL FOR HOST PROCESSOR AND UTS 400

3.5.1. Rules for Basic Communication Between Host Processor and UTS 400

The following rules define the basic communication procedures between the host processor and the UTS 400. These rules are based on the assumption that no peripherals are involved, only one station in the poll group is being used, and the program attention keys are not being used.

- **UTS 400 Rule 1:**

  The station responds only to error-free polls.

- **UTS 400 Rule 2:**

  The station expects an acknowledgment to any message it sends in response to a poll except the no-traffic message.

- **UTS 400 Rule 3:**

  A station will not send two consecutive text messages. Thus, a station does not respond with a text message to a poll that includes the acknowledgment to the previous text message.

- **UTS 400 Rule 4:**

  The station acknowledges in its next poll response any error-free host processor message containing an STX (text message, message wait command, or disconnection command).

- **Host Processor Rule 1:**

  Upon sending a text message to a station, the host processor must poll to verify proper receipt of the text. This poll must occur before any other text is sent to that poll group.

- **Host Processor Rule 2:**

  The host processor must send a poll with acknowledgment to a station that has sent an acknowledgeable response before the host processor can send a text message to that station.

3.5.2. UTS 400 Text Message Example Sequence

The following message sequence applies UTS 400 rules 1, 2, and 3.

In this sequence, it is assumed that the operator has pressed the transmit key on a station whose RID is 1 and SID is a.

1.	**SOH 1 P p ETX BCC**	$\longrightarrow$	
2.		$\longleftarrow$	**SOH 1 a p STX data ETX BCC**
3.	**SOH 1 P p DLE 1 ETX BCC**	$\longrightarrow$	
4.		$\longleftarrow$	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be discussed.

1. The host processor sends a general traffic poll to a poll group whose RID is 1.

2. The poll response is UTS 400 text from the station whose RID is 1 and SID is a. (See UTS 400 rule 1.)

3. The host processor sends a general traffic poll with acknowledgment (DLE 1). This signifies to the station that the host processor correctly received its text message; that is, the UTS 400 text message did not contain any characters with incorrect parity and the BCC in the UTS 400 text message matched the BCC accumulated by the host processor. (See UTS 400 rule 2.)

4. The station cannot send another text message at this time; therefore, the station's response must be a no-traffic message. (See UTS 400 rule 3.)

3.5.3. Host Processor Text Message Example Sequence

The following message sequence applies UTS 400 rules 1, 2, and 4 and host processor rules 1 and 2.

1.	**SOH 1 a p STX data ETX BCC**	$\longrightarrow$	
2.	**SOH 1 P p ETX BCC**	$\longrightarrow$	
3.		$\longleftarrow$	**SOH 1 a p DLE 1 ETX BCC**
4.	**SOH 1 a p DLE 1 ETX BCC**	$\longrightarrow$	
5.		$\longleftarrow$	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be discussed.

1. The host processor sends a text message to a station whose RID is 1 and SID is a. This station must not be expecting an acknowledgment; that is, its previous poll response (assuming it is the only station in the poll group) would have been a no-traffic message.

2. The host processor sends a general traffic poll to verify that the station received the text message correctly. (See host processor rule 1.)

3. The acknowledgment (DLE 1) indicates that the station correctly received the text message from the host processor; that is, the text message contained no characters with incorrect parity and the BCC in the text message matched the BCC accumulated by the UTS 400. (See UTS 400 rule 4.)

4. The host processor must acknowledge any UTS 400 message other than a no-traffic message. (See host processor rule 2.)

5. The UTS 400 correctly received the poll-with-acknowledgment message from the host processor and responds with a no-traffic message.

3.6. MULTIPLEXER FUNCTIONS

A terminal multiplexer, a UTS 400 master, or a UTS 400 controller each has the capability to determine which station (under its control) in a poll group will respond to a poll. The multiplexer function allows responses within the poll group to be sent in the following priority sequence:

1. A station that expected an acknowledgment but did not receive it is allowed to respond with a reply request.

2. If the priority 1 conditions do not exist, a station with traffic is allowed to respond. The response may include an acknowledgment (ACK or WABT) from that station or an ACK/WABT passed from another station within that poll group, thus creating a composite message containing ACK or WABT plus traffic.

3. If the priority 2 conditions do not exist, a station with an outstanding ACK or WABT is allowed to respond.

4. If the priority 3 conditions do not exist, one of the stations that has no traffic is allowed to respond to a host processor poll.

3.6.1. Rules for Handling Messages Modified by the Multiplexer Function

The following rules define the message flow for communication between the host processor and the UTS 400 as modified by the multiplexer function of a terminal multiplexer, a UTS 400 master, or a UTS 400 controller.

- **Host Processor Rule 3:**

  When using general polls, the host processor must expect an acknowledgment (ACK or WABT) from one station to be included with a response from another station in the poll group. The multiplexer function allows an acknowledgment to be passed from one station to another station that has a traffic response pending.

- **Host Processor Rule 4:**

  When using general polls; the host processor must expect successive traffic responses from stations within a poll group. However, UTS 400 rule 3 still applies. The multiplexer function allows a station with a message pending to send its response to any poll whose address it recognizes.

- **Host Processor Rule 5:**

  The host processor can send text to any station in the poll group that is not owed an acknowledgment, provided the last response from the station is not one of the following:

  – ACK only (SOH RID SID DID DLE 1 ETX BCC)

  – WABT only (SOH RID SID DID DLE ? ETX BCC)

  – ACK plus text available (SOH RID SID DID DLE 1 DLE 0 ETX BCC)

  – WABT plus text available (SOH RID SID DID DLE ? DLE 0 ETX BCC)

- **Host Processor Rule 6:**

  When the host processor owes an acknowledgment to a station in a poll group, the host processor may send a specific poll to the poll group only if the specific poll is addressed to the station that is owed the acknowledgment.

- **Host Processor Rule 7:**

  The host processor must not allow a given poll group to owe more than one acknowledgment to the host processor at any time.

3.6.2. Example Message Sequence Involving the Multiplexer Function

Example Sequence 1:

The following message sequence applies the multiplexer function definitions and host processor rules 3, 4, and 5.

In this sequence, it is assumed that the poll group defined by the UTS 400 master/slave cluster shown in Figure 3–1 is being used.

1.	SOH 1 P p ETX BCC	→	
2.		←	SOH 1 a p STX data ETX BCC
3.	SOH 1 P p DLE 1 ETX BCC	→	
4.		←	SOH 1 b p STX data ETX BCC
5.	SOH 1 P p DLE 1 ETX BCC	→	
6.		←	EOT EOT ETX BCC
7.	SOH 1 a p STX data ETX BCC	→	
8.	SOH 1 P p ETX BCC	→	
9.		←	SOH 1 c p DLE 1 STX data ETX BCC
10.	SOH 1 P p DLE 1 ETX BCC	→	
11.		←	EOT EOT ETX BCC

Each message in the preceding sequence will now be explained.

1. General traffic poll to the poll group whose RID is 1.
2. UTS 400 text from the station whose RID is 1 and SID is a.
3. General traffic poll with acknowledgment.
4. UTS 400 text from the station whose RID is 1 and SID is b. (See host processor rule 4.)
5. General traffic poll with acknowledgment.
6. No traffic.
7. Host processor text to the station whose RID is 1 and SID is a. (See host processor rule 5.)
8. General traffic poll.
9. Acknowledgment plus traffic where the traffic is text. The text is from the station whose RID is 1 and SID is c. The acknowledgment was passed by the UTS 400 master from the station whose RID is 1 and SID is a. (See host processor rule 3.)
10. General traffic poll with acknowledgment.
11. No traffic.

Example Sequence 2:

The following message sequence applies the multiplexer function definitions and host processor rules 3, 4, and 5.

In this sequence, it is assumed that the poll group used is the master/slave cluster shown in Figure 3–1.

1.	**SOH 1 P p ETX BCC**	→	
2.		←	**SOH 1 a p STX data ETX BCC**
3.	**SOH 1 P p DLE 1 ETX BCC**	→	
4.		←	**SOH 1 b p STX data ETX BCC**
5.	**SOH 1 a p STX data ETX BCC**	→	
6.	**SOH 1 P p DLE 1 ETX BCC**	→	
7.		←	**SOH 1 c p DLE 1 STX data ETX BCC**
8.	**SOH 1 b p STX data ETX BCC**	→	
9.	**SOH 1 P p DLE 1 ETX BCC**	→	
10.		←	**SOH 1 b p DLE 1 ETX BCC**

```
11.  SOH  1  P  p  DLE 1  ETX  BCC       ────────►
12.                                      ◄────────   EOT  EOT  ETX  BCC
13.  SOH  1  c  p  STX  data  ETX  BCC   ────────►
14.  SOH  1  P  p  ETX  BCC               ────────►
15.                                      ◄────────   SOH  1  c  p  DLE 1  ETX  BCC
16.  SOH  1  P  p  DLE 1  ETX  BCC       ────────►
17.                                      ◄────────   EOT  EOT  ETX  BCC
```

Each message in the preceding sequence will now be explained.

1. General traffic poll to the poll group whose RID is 1.
2. UTS 400 text from the station whose RID is 1 and SID is a.
3. General traffic poll with acknowledgment.
4. UTS 400 text from the station whose RID is 1 and SID is b.
5. Host processor text to the station whose RID is 1 and SID is a. (See host processor rule 5.)
6. General traffic poll with acknowledgment.
7. Acknowledgment plus traffic where the traffic is text. The text is from the station whose RID is 1 and SID is c. The acknowledgment was passed by the UTS 400 master from the station whose RID is 1 and SID is a. (See host processor rule 3.)
8. Host processor text to the station whose RID is 1 and SID is b. (See host processor rule 5.)
9. General traffic poll with acknowledgment.
10. Acknowledgment from the station whose RID is 1 and SID is b.
11. General traffic poll with acknowledgment. Text cannot be sent to the poll group at this point because the last response from the poll group was ACK (DLE 1). (See host processor rule 5.)
12. No traffic.
13. Host processor text to the station whose RID is 1 and SID is c. (See host processor rule 5.)
14. General traffic poll.
15. Acknowledgment from the station whose RID is 1 and SID is c.

16. General traffic poll with acknowledgment.

17. No traffic.

NOTE:

If the RID value is changed, the preceding two examples are equally valid for the poll group defined by the UTS 400 controller/slave cluster or the poll group defined by the three UTS 400 masters connected to a terminal multiplexer as shown in Figure 3–1.

3.7. COMMUNICATIONS LINE ERROR RECOVERY

3.7.1. Rules for Line Error Recovery

Transmission errors may occur in messages to and from the UTS 400. The following rules define the line error recovery procedures to be followed in communication between the host processor and the UTS 400.

■ **UTS 400 Rule 5:**

  If a station that is owed an acknowledgment does not receive an acknowledgment in the next good poll that it recognizes, the station sends a reply request (DLE ENQ) message.

■ **UTS 400 Rule 6:**

  A station will not send an acknowledgment (ACK or WABT) with a reply request. A station having a reply request can be passed an acknowledgment because of the multiplexer function. However, that acknowledgment will not be reported until the reply request condition is satisfied.

■ **Host Processor Rule 8:**

  The host processor must treat any error in a received message as a no-response condition and must repeat the poll that preceded the no-response condition. However, any acknowledgment included with the original poll must be eliminated and a general DID must be used when the poll is repeated. If the no-response condition results from a retransmission request (DLE NAK), the host processor must repeat the poll that created the reply request response.

■ **Host Processor Rule 9:**

  The host processor response to a reply request (DLE ENQ) from a station must be a retransmission request (DLE NAK) if the last message correctly received from the station sending the reply request was one not containing text data. The retransmission request has the same specific RID and specific SID address as that contained in the reply request. The retransmission request must contain a general DID.

■ **Host Processor Rule 10:**

  The host processor response to a reply request (DLE ENQ) from a station must be a poll-with-acknowledgment message if the last message correctly received from the station sending the reply request was a message whose traffic was text. The acknowledgment is for the message that preceded the reply request and not for the reply request message itself.

- **Host Processor Rule 11:**

  The host processor response to a reply request (DLE ENQ) from a station must be a retransmission request (DLE NAK) if the station sending the reply request is other than one to which an acknowledgment has just been sent.

- **Host Processor Rule 12:**

  If the response from a retransmission request is identical to the UTS 400 response sent just previous to the reply request and is from the same station, then the response to the retransmission request can be ignored except for sending the acknowledgment that the station expects. However, the same UTS 400 response is to be ignored only once.

3.7.2. Example Line Error Message Sequence Involving Polls

The following message sequence applies host processor rule 8.

1. **SOH 1 P p ETX BCC** $\longrightarrow$
2. No-response condition
3. **SOH 1 P p ETX BCC** $\longrightarrow$
4. $\longleftarrow$ **EOT EOT ETX BCC**

Each message in the preceding message sequence will now be explained.

1. General traffic poll.
2. No-response condition indicates transmission error occurred on poll or response.
3. The general traffic poll sent in step 1 is repeated. (See host processor rule 8.)
4. No-traffic response. Error recovery was successful.

3.7.3. Example Line Error Message Sequences Involving UTS 400 Text

Example Sequence 1:

The following message sequence applies UTS 400 rule 5 and host processor rules 8 and 9.

1. **SOH 1 P p ETX BCC** $\longrightarrow$
2. $\longleftarrow$ **EOT EOT ETX BCC**
3. **SOH 1 P p ETX BCC** $\longrightarrow$
4. No-response condition
5. **SOH 1 P p ETX BCC** $\longrightarrow$

6.		←	SOH 1 a p DLE ENQ ETX BCC
7.	SOH 1 a p DLE NAK ETX BCC	→	
8.		←	SOH 1 a p STX data ETX BCC
9.	SOH 1 P p DLE 1 ETX BCC	→	
10.		←	EOT EOT ETX BCC

Each message in the preceding sequence will now be explained.

1. General traffic poll.
2. No-traffic response.
3. General traffic poll.
4. No-response condition indicates transmission error occurred on poll or response.
5. General traffic poll is repeated. (See host processor rule 8.)
6. Reply request from the station whose RID is 1 and SID is a. (See UTS 400 rule 5.)
7. Retransmission request sent to the station that sent the reply request in step 6. Host processor rule 9 applies because the UTS 400 message (step 2) just previous to the reply request did not contain text data.
8. UTS 400 text from the station whose RID is 1 and SID is a.
9. General traffic poll with acknowledgment.
10. No traffic.

Example Sequence 2:

The following message sequence applies UTS 400 rule 5 and host processor rules 8 and 9.

1.	SOH 1 P p ETX BCC	→	
2.		←	SOH 1 a p STX data ETX BCC
3.	SOH 1 P p DLE 1 ETX BCC	→	
4.		No-response condition	
5.	SOH 1 P p ETX BCC	→	
6.		←	SOH 1 a p DLE ENQ ETX BCC
7.	SOH 1 P p DLE 1 ETX BCC	→	
8.		←	EOT EOT ETX BCC

Each message in the preceding sequence will now be explained.

1. General traffic poll.
2. UTS 400 text from the station whose RID is 1 and SID is a.
3. General traffic poll with acknowledgment.
4. No-response condition indicates transmission error occurred on poll or response.
5. General traffic poll is repeated. (See host processor rule 8.)
6. Reply request from the station whose RID is 1 and SID is a. (See UTS 400 rule 5.)
7. A general traffic poll with acknowledgment is sent to the same station that just sent the reply request. (See host processor rule 10.)
8. No-traffic response. Error recovery was successful.

Example Sequence 3:

The following message sequence applies UTS 400 rule 5 and host processor rules 8 and 11.

1.	**SOH 1 P p ETX BCC**	→	
2.		←	**SOH 1 a p STX data ETX BCC**
3.	**SOH 1 P p DLE 1 ETX BCC**	→	
4.		No-response condition	
5.	**SOH 1 P p ETX BCC**	→	
6.		←	**SOH 1 b p DLE ENQ ETX BCC**
7.	**SOH 1 b p DLE NAK ETX BCC**	→	
8.		←	**SOH 1 b p STX data ETX BCC**
9.	**SOH 1 P p DLE 1 ETX BCC**	→	
10.		←	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained

1. General traffic poll.
2. UTS 400 text from the station whose RID is 1 and SID is a.

3. General traffic poll with acknowledgment.
4. No-response condition indicates transmission error occurred on poll or response.
5. General traffic poll is repeated. (See host processor rule 8.)
6. Reply request from the station whose RID is 1 and SID is b. (See UTS 400 rule 5.)
7. A retransmission request must be sent as described in host processor rule 11.
8. UTS 400 text from the station whose RID is 1 and SID is b.
9. General traffic poll with acknowledgment.
10. No traffic.

3.7.4. Example Line Error Message Sequences Involving Host Processor Text

Example Sequence 1:

The following message sequence applies UTS 400 rule 4 and host processor rule 5.

1.	**SOH 1 a p STX data ETX BCC**	⟶	
2.	**SOH 1 P p ETX BCC**	⟶	
3.		⟵	**EOT EOT ETX BCC**
4.	**SOH 1 a p STX data ETX BCC**	⟶	
5.	**SOH 1 P p ETX BCC**	⟶	
6.		⟵	**SOH 1 a p DLE 1 ETX BCC**
7.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
8.		⟵	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. Host processor text to the station whose RID is 1 and SID is a.
2. General traffic poll.
3. The no-traffic response indicates either that the host processor text message sent in step 1 was not error free when it arrived at the UTS 400 or that the station was not connected and another station in the poll group furnished the no-traffic response. The host processor must resend this text message. (See UTS 400 rule 4.)
4. The host processor text message in step 1 is resent. (See host processor rule 5.)

5. General traffic poll.

6. The acknowledgment indicates that the host processor text sent in step 4 was received error free. (See UTS 400 rule 4.)

7. General traffic poll with acknowledgment.

8. No traffic.

Example Sequence 2:

The following message sequence applies UTS 400 rule 4 and host processor rule 5.

1.	**SOH 1 a p STX data ETX BCC**	⟶	
2.	**SOH 1 P p ETX BCC**	⟶	
3.		⟵	**SOH 1 b p STX data ETX BCC**
4.	**SOH 1 a p STX data ETX BCC**	⟶	
5.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
6.		⟵	**SOH 1 a p DLE 1 ETX BCC**
7.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
8.		⟵	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. Host processor text to the station whose RID is 1 and SID is a.

2. General traffic poll.

3. The absence of an acknowledgment in the station response indicates either that the host processor text message in step 1 was not error free when it arrived at the UTS 400 or that the station was not connected. The host processor must resend this text message. (See UTS 400 rule 4.)

4. The host processor text message in step 1 is resent. (See host processor rule 5.)

5. General traffic poll with acknowledgment.

6. The acknowledgment indicates that the host processor text sent in step 4 was received error free. (See UTS 400 rule 4.)

7. General traffic poll with acknowledgment.

8. No traffic.

Example Sequence 3:

The following message sequence applies UTS 400 rule 5 and host processor rules 8, 9, and 12.

1.	SOH 1 a p STX data ETX BCC	$\longrightarrow$	
2.	SOH 1 P p ETX BCC	$\longrightarrow$	
3.		No-response condition	
4.	SOH 1 P p ETX BCC	$\longrightarrow$	
5.		$\longleftarrow$	SOH 1 a p DLE 1 ETX BCC
6.	SOH 1 P p DLE 1 ETX BCC	$\longrightarrow$	
7.		No-response condition	
8.	SOH 1 P p ETX BCC	$\longrightarrow$	
9.		$\longleftarrow$	SOH 1 a p DLE ENQ ETX BCC
10.	SOH 1 a p DLE NAK ETX BCC	$\longrightarrow$	
11.		$\longleftarrow$	SOH 1 a p DLE 1 ETX BCC
12.	SOH 1 P p DLE 1 ETX BCC	$\longrightarrow$	
13.		$\longleftarrow$	EOT EOT ETX BCC

Each message in the preceding sequence will now be explained.

1. Host processor text to the station whose RID is 1 and SID is a.

2. General traffic poll.

3. No-response condition indicates transmission error occurred on poll or response.

4. General traffic poll is repeated.

5. Acknowledgment from the station that was sent text in step 1.

6. General traffic poll with acknowledgment.

7. No-response condition indicates transmission error occurred on poll or response.

8. The poll in step 6 is repeated without the ACK (DLE 1). (See host processor rule 8.)

9. Reply request (DLE ENQ) from the station whose RID is 1 and SID is a.

10. Retransmission request (DLE NAK) to the station that sent the reply request in step 9. Host processor rule 9 applies because the UTS 400 message (step 5) sent previous to the reply request did not include text data.

11. This acknowledgment is extraneous, as described in host processor rule 12. It is a repeat of the acknowledgment successfully received in step 5.

12. General traffic poll with acknowledgment.

13. No traffic.

Example Sequence 4:

The following message sequence applies UTS 400 rule 5 and host processor rules 8 and 9.

1.	**SOH 1 P p ETX BCC**	→	
2.		←	**EOT EOT ETX BCC**
3.	**SOH 1 a p STX data ETX BCC**	→	
4.	**SOH 1 P p ETX BCC**	→	
5.		No-response condition	
6.	**SOH 1 P p ETX BCC**	→	
7.		←	**SOH 1 a p DLE ENQ ETX BCC**
8.	**SOH 1 a p DLE NAK ETX BCC**	→	
9.		←	**SOH 1 a p DLE 1 ETX BCC**
10.	**SOH 1 P p DLE 1 ETX BCC**	→	
11.		←	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. General traffic poll.

2. No-traffic response.

3. Host processor text to the station whose RID is 1 and SID is a.

4. General traffic poll.

5. No-response condition indicates transmission error occurred on poll or response.

6. General traffic poll is repeated.

7. Reply request (DLE ENQ) from the station whose RID is 1 and SID is a.

8. Retransmission request (DLE NAK) to the station that sent the reply request in step 7. Host processor rule 9 applies because the UTS 400 message (step 2) sent previous to the reply request did not include text data.

9. This acknowledgment is a retransmission of the UTS 400 message that was not error free in step 5.

10. General traffic poll with acknowledgment.

11. No traffic.

Example Sequence 5:

The following message sequence applies UTS 400 rule 5 and host processor rules 5, 8, and 11.

1.	**SOH 1 P p ETX BCC**	$\longrightarrow$	
2.		$\longleftarrow$	**SOH 1 a p STX data ETX BCC**
3.	**SOH 1 b p STX data ETX BCC**	$\longrightarrow$	
4.	**SOH 1 P p DLE 1 ETX BCC**	$\longrightarrow$	
5.		No-response condition	
6.	**SOH 1 P p ETX BCC**	$\longrightarrow$	
7.		$\longleftarrow$	**SOH 1 b p DLE ENQ ETX BCC**
8.	**SOH 1 b p DLE NAK ETX BCC**	$\longrightarrow$	
9.		$\longleftarrow$	**SOH 1 b p DLE 1 ETX BCC**
10.	**SOH 1 P p DLE 1 ETX BCC**	$\longrightarrow$	
11.		$\longleftarrow$	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. General traffic poll.

2. UTS 400 text message from the station whose RID is 1 and SID is a.

3. Host processor text directed to the station whose RID is 1 and SID is b. (See host processor rule 5.)

4. General traffic poll with acknowledgment.

5. No-response condition indicates transmission error occurred on poll or response.

6. General traffic poll is repeated. (See host processor rule 8.)

7. Reply request from the station whose RID is 1 and SID is b. (See UTS 400 rule 5.)

8. Retransmission request to the station that sent the reply request in step 7. Host processor rule 11 applies because an acknowledgment was just sent for the station whose RID is 1 and SID is a (steps 2 and 4), but the reply request was from the station whose RID is 1 and SID is b.

9. Acknowledgment for the text message in step 3.

10. General poll with acknowledgment.

11. No traffic.

3.7.5. Example Line Error Message Sequences Involving Passing the Acknowledgment

Example Sequence 1:

The following message sequence applies UTS 400 rule 5 and host processor rules 8 and 9.

1.	SOH 1 P p ETX BCC	⟶	
2.		⟵	EOT EOT ETX BCC
3.	SOH 1 a p STX data ETX BCC	⟶	
4.	SOH 1 P p ETX BCC	⟶	
5.		No-response condition	
6.	SOH 1 P p ETX BCC	⟶	
7.		⟵	SOH 1 b p DLE ENQ ETX BCC
8.	SOH 1 b p DLE NAK ETX BCC	⟶	
9.		⟵	SOH 1 b p DLE 1 STX data ETX BCC
10.	SOH 1 P p DLE 1 ETX BCC	⟶	
11.		⟵	EOT EOT ETX BCC

Each message in the preceding sequence will now be explained.

1. General traffic poll.

2. No traffic.

3. Host processor text message directed to the station whose RID is 1 and SID is a.

4. General traffic poll.

5. No-response condition indicates transmission error occurred on poll or response.

6. The general traffic poll must be repeated. (See host processor rule 8.)

7. Reply request (DLE ENQ) from the station whose RID is 1 and SID is b. (See UTS 400 rule 5.)

8. Retransmission request (DLE NAK) to the station that sent the reply request in step 7. Host processor rule 9 applies because the UTS 400 message (step 2) sent previous to the reply request was a no-traffic message.

9. Acknowledgment-plus-traffic message, where the traffic is text. The station had sent this same message in step 5, but it was in error. The acknowledgment (DLE 1) in the message was passed by the multiplexer function from the station whose RID is 1 and SID is a (the one to which text was sent in step 3).

10. General traffic poll with acknowledgment.

11. No traffic.

Example Sequence 2:

The following message sequence applies UTS 400 rules 5 and 6 and host processor rules 5, 8, and 10.

1.	**SOH 1 P p ETX BCC**	⟶	
2.		⟵	**SOH 1 a p STX data ETX BCC**
3.	**SOH 1 b p STX data ETX BCC**	⟶	
4.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
5.		No-response condition	
6.	**SOH 1 P p ETX BCC**	⟶	
7.		⟵	**SOH 1 a p DLE ENQ ETX BCC**
8.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
9.		⟵	**SOH 1 a p DLE 1 ETX BCC**
10.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
11.		⟵	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. General traffic poll.

2. UTS 400 text from the station whose RID is 1 and SID is a.

3. Host processor text to the station whose RID is 1 and SID is b. (See host processor rule 5.)

4. General traffic poll with acknowledgment.

5. No-response condition indicates transmission error occurred on poll or response.

6. The general traffic poll must be repeated. (See host processor rule 8.)

7. Reply request (DLE ENQ) message from the station whost RID is 1 and SID is a.

8. General traffic poll with acknowledgment. (See host processor rule 10.)

9. Acknowledgment for the text message in step 3. The acknowledgment (DLE 1) was passed by the multiplexer function (from the station whose RID is 1 and SID is b to the station whose RID is 1 and SID is a) at the time the general traffic poll (step 6) was received by the UTS 400. However, this acknowledgment was not sent with the reply request in step 7 (UTS 400 rule 6).

10. General traffic poll with acknowledgment.

11. No traffic.

Example Sequence 3:

The following message sequence applies UTS 400 rules 5 and 6 and host processor rules 5, 8, and 10.

1.	**SOH 1 P p ETX BCC**	→	
2.		←	**EOT EOT ETX BCC**
3.	**SOH 1 a p STX data ETX BCC**	→	
4.	**SOH 1 P p ETX BCC**	→	
5.		←	**SOH 1 c p DLE 1 STX data ETX BCC**
6.	**SOH 1 b p STX data ETX BCC**	→	
7.	**SOH 1 P p DLE 1 ETX BCC**	→	
8.		No-response condition	
9.	**SOH 1 P p ETX BCC**	→	
10.		←	**SOH 1 c p DLE ENQ ETX BCC**
11.	**SOH 1 P p DLE 1 ETX BCC**	→	
12.		←	**SOH 1 c p DLE 1 ETX BCC**
13.	**SOH 1 P p DLE 1 ETX BCC**	→	
14.		←	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. General traffic poll.
2. No traffic.
3. Host processor text to the station whose RID is 1 and SID is a.
4. General traffic poll.
5. Acknowledgment plus traffic, where the traffic is text. The text is from the station whose RID is 1 and SID is c. The acknowledgment was passed by the multiplexer function from the station whose RID is 1 and SID is a.
6. Host processor text to the station whose RID is 1 and SID is b. (See host processor rule 5.)
7. General traffic poll with acknowledgment.
8. No-response condition indicates transmission error occurred on poll or response.
9. General traffic poll is repeated. (See host processor rule 8.)
10. Reply request from the station whose RID is 1 and SID is c. (See UTS 400 rule 5.)
11. General traffic poll with acknowledgment. Host processor rule 10 applies because the last response (step 5) from the poll group contained text from the same station that sent the reply request in step 10.
12. Acknowledgment for the text message sent in step 6. The acknowledgment was passed by the multiplexer function (from the station whose RID is 1 and SID is b to the station whose RID is 1 and SID is c) at the time the general traffic poll (step 9) was received by the UTS 400. However, this acknowledgment was not sent with the reply request in step 10 (UTS 400 rule 6).
13. General traffic poll with acknowledgment.
14. No traffic.

3.7.6. Example Line Error Message Sequence Involving Errors During Recovery

The following message sequence applies UTS 400 rule 5 and host processor rules 8 and 9.

1.	**SOH 1 P p ETX BCC**	⟶	
2.		⟵	**EOT EOT ETX BCC**
3.	**SOH 1 P p ETX BCC**	⟶	
4.		No-response condition	
5.	**SOH 1 P p ETX BCC**	⟶	

6.		←	**SOH 1 a p DLE ENQ ETX BCC**
7.	**SOH 1 a p DLE NAK ETX BCC**	→	
8.		No-response condition	
9.	**SOH 1 a p ETX BCC**	→	
10.		←	**SOH 1 a p DLE ENQ ETX BCC**
11.	**SOH 1 a p DLE NAK ETX BCC**	→	
12.		←	**SOH 1 a p STX data ETX BCC**
13.	**SOH 1 P p DLE 1 ETX BCC**	→	
14.		←	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. General traffic poll.
2. No traffic.
3. General traffic poll.
4. No-response condition indicates transmission error occurred on poll or response.
5. General traffic poll is repeated.
6. Reply request (DLE ENQ) from the station whose RID is 1 and SID is a. (See UTS 400 rule 5.)
7. Retransmission request (DLE NAK) to the station that sent the reply request in step 6. Host processor rule 9 applies because the UTS 400 message (step 2) previous to the reply request did not include text data.
8. No-response condition indicates transmission error occurred on poll or response.
9. A specific traffic poll is sent to the same station to which a retransmission request was sent in step 7. (See host processor rule 8.)
10. Repeat of the reply request from the station whose RID is 1 and SID is a.
11. Retransmission request of step 7 is repeated.
12. This UTS 400 text message is the same as the one on which a line error occurred in step 4.
13. General traffic poll with acknowledgment.
14. No traffic.

3.8. PERIPHERAL OPERATION

→ The host processor can access the UTS 400 share-type peripherals through the screen of a UTS 400 station or through the optional screen bypass feature. All stations and the screen bypass must wait in a queue for access to the peripheral devices.

The peripheral-sharing capability of the UTS 400 introduces several conditions that existing communications handlers may not recognize. One such condition is the UTS 400 capability for automatic retry of peripheral operations. When automatic peripheral retry is enabled (by means of a ROM switch), the UTS 400 will twice retry certain peripheral operations after an error condition occurs (for example, a tape cassette system data error) before reporting any indication of an operation failure.

NOTE:

Automatic peripheral retry must be enabled when multiple stations are sharing peripherals and those peripherals are subject to host-initiated activity.

The UTS 400 accepts several peripheral initiation commands, which are discussed in 4.6.1. In the remainder of this section, the term "PI" is used to represent any one of the peripheral initiation commands. Other representative terms used in this section are:

TEXT	Represents a host processor text message containing a general DID and no peripheral initiation command.
TEXT/SD	Represents a host processor text message containing a specific DID (SD) and no peripheral initiation command.
TEXT/PI	Represents a host processor text message containing a general DID and a peripheral initiation (PI) command. The peripheral must have been previously selected by the host processor before the TEXT/PI message can be sent by the host processor.
TEXT/SD/PI	Represents a host processor text message containing a specific DID (SD) and a peripheral initiation (PI) command.

3.8.1. Peripheral Access Queue (PAQ)

The stations that need access to the peripheral interface are queued by the UTS 400 master or controller. Henceforth, this queue will be called the "peripheral access queue," or PAQ. The station at the "top of the PAQ" is the station that has current access to the peripheral interface.

Stations can be placed in the PAQ by the UTS 400 master or controller when:

- A selection poll is received from the host processor; or
- A TEXT/SD message is received from the host processor; or
- A TEXT/PI message is received from the host processor; or
- A TEXT/SD/PI message is received from the host processor; or
- The station operator presses the PRINT key, XFER key, SEARCH key, REP ADR key, BOB key, or STATUS key.

The data portion of any text message from the host processor is stored in the CRT refresh memory of the station or in the screen bypass memory, as specified by the SID, and is maintained there even if the transaction is placed in the PAQ. Thus, even though a station may not be at the top of the PAQ, the message is stored in memory and appears immediately on the screen of that station.

Host processor messages are always buffered in the CRT refresh memory or in the screen bypass memory. However, messages received as a result of a station operator's pressing the PRINT key or XFER key may be buffered in the CRT refresh memory or in optional random-access memory (RAM) designated as a peripheral buffer pool area.

3.8.2. UTS 400 Notification to the Host Processor of PAQ Status

A station gains immediate access to the peripheral interface whenever the PAQ is empty. If the host processor message is a TEXT/PI or TEXT/SD/PI message, the normal poll response from the station is a message containing the WABT (DLE ?) acknowledgment sequence. (The WABT message format is shown in 3.4.2.3.) The WABT indicates that an error-free message was received by the station and that the peripheral operation was initiated. A later poll response containing the THRU (DLE ;) sequence indicates completion of the peripheral operation. (The THRU message format is shown in 3.4.2.4.4.)

When a message sent from the host processor to a station causes that station to be placed in the PAQ, and there are already other stations in the PAQ, the station returns one of the following station status messages in a poll response:

- Peripheral-selection-delayed status (DLE 5) if the host processor message is a selection poll
- ACK plus peripheral-selection-delayed status (DLE 1 DLE 5) if the host processor message is TEXT/SD
- ACK plus message-queued status (DLE 1 DLE 4) if the host processor message is TEXT/PI or TEXT/SD/PI

The formats for these UTS 400 station status messages are shown in 3.4.2.4.4.

3.8.3. DID Address Functions

The host processor selects a UTS 400 share-type peripheral by sending a specific DID. The communications output printer (COP), 800 terminal printer, and 0786 printer are addressed by a single DID each, and the tape cassette system and diskette subsystem are addressed by two DIDs per drive (one read DID and one write DID).

Peripheral selections are performed with the following types of host processor messages:

- Selection poll
- TEXT/SD message
- TEXT/SD/PI message

Once a peripheral is selected (by means of a specific DID) for a particular station addressed by the RID SID, the peripheral remains associated with that station (SID) for subsequent host messages for peripherals. The same peripheral can be utilized by more than one station in a UTS 400 cluster. However, the user is responsible for maintaining separation of files and/or forms on the peripheral. The peripheral selection is maintained at the station level by the UTS 400 master or the UTS 400 controller.

If a UTS 400 that does not contain the peripheral interfaces receives a selection poll, the response is a no-traffic message. In the same situation, the response to a poll following a TEXT/SD message or a TEXT/SD/PI message is the ACK (DLE 1) response.

Peripheral status responses from UTS 400 systems with peripheral interfaces are discussed in 3.8.4.

The way in which a host processor message for a peripheral is handled by the UTS 400 master or controller is determined by the DID value in the message. DID values are interpreted as follows:

Value (octal)	Value (hexadecimal)	
160	70	This is the general DID. It has no effect on the peripheral interface or on the peripherals.
163 thru 176	73 thru 7E	These are the 12 DIDs available for use by the host processor as peripheral addresses.
161	71	This DID value is used to override peripheral activity. Any station receiving this DID will terminate both active and queued peripheral operations. No peripheral status will be sent; however, a THRU station status may be returned as a response to this poll (refer to Appendix C).
162	72	This is the deselection DID. A deselection has special impact on the tape cassette system (refer to Appendix C).
177	7F	This DID is included in UTS 400 poll responses if the station PRINT key or XFER key has been pressed.

3.8.4. Rules for Host Processor/UTS 400 Peripheral Communication

The following rules define communication procedure between the host processor and the UTS 400 share-type peripherals.

- **UTS 400 Rule 7:**

  The specific DID in any host processor message causes a selection attempt only if the message is error free.

- **UTS 400 Rule 8:**

  The UTS 400 will respond to a selection poll or to the first poll following a TEXT/SD, TEXT/SD/PI, or TEXT/PI message with a peripheral status response. These peripheral status responses are listed in Table 3–7.

- **UTS 400 Rule 9:**

  The UTS 400 will not respond to a selection poll or to a TEXT/SD/PI message when an offline peripheral buffering operation is in progress. This condition appears to the host processor as if an error occurred on the poll or response. The normal no-response error recovery should be invoked.

- **UTS 400 Rule 10:**

  A selection poll requiring access to a peripheral interface already in use by another station will solicit the peripheral-selection-delayed (DLE 5) response. The station for which the selection is to be performed is added to the PAQ. When that station reaches the top of the PAQ, the selection attempt is made and the next general poll or specific poll response from that station will be DLE > (peripheral ready), DLE <, DLE :, or DLE = (no response from the peripheral). The interpretations of DLE < and DLE : depend on the peripheral in use.

Table 3–7. UTS 400 Peripheral Status Responses to Host Processor Messages

Peripheral Condition	Selection Poll From Host Processor	TEXT/SD From Host Processor	TEXT/SD/PI From Host Processor	TEXT/PI From Host Processor (after previous selection)
No peripheral interface support within UTS 400	No traffic	DLE 1	DLE 1	DLE 1
Peripheral response 1 (ready)	DLE >	DLE 1 DLE >	DLE ? or DLE 1 DLE ; (notes 1 and 3)	DLE ? or DLE 1 DLE ; (notes 1 and 3)
Peripheral response 2	DLE <	DLE 1 DLE <	DLE 1 DLE < or DLE ? (note 2)	DLE ?
Peripheral response 3	DLE :	DLE 1 DLE :	DLE 1 DLE : or DLE ? (note 2)	DLE ?
No peripheral response	DLE =	DLE 1 DLE =	DLE 1 DLE =	DLE ?
Peripheral selection delayed (another station using the peripheral interface)	DLE 5	DLE 1 DLE 5	DLE 1 DLE 4 (note 3)	DLE 1 DLE 4 (note 3)

NOTES:

1. 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 host processor.

2. The response depends on the peripheral. An acknowledgment plus error status implies the peripheral initiation did not occur. The busy status implies the initiation was attempted. If the peripheral initiation was successful, the THRU status will subsequently be reported; if unsuccessful, a sustained busy will result. For example, if an end-of-tape condition exists on a tape cassette, an attempt to write would result in the acknowledgment plus end-of-tape status (peripheral status 2) response, an attempt to read would result in a sustained busy status, and an attempt to rewind would be successful (result in a THRU status).

3. In a multistation environment, the DLE 1 portion of the message may be sent independently of the status portion when a DLE 1 plus status condition exists (refer to 3.11.2).

- **UTS 400 Rule 11:**

  A TEXT/PI message or a TEXT/SD/PI message requires access to the peripheral interface. If the peripheral interface is already in use, the station response is ACK plus message queued (DLE 1 DLE 4). The station response to subsequent general polls is no traffic, and the response to subsequent specific polls is message queued (DLE 4) until the message reaches the top of the PAQ. The station then responds to subsequent specific polls with a WABT (DLE ?) as long as the peripheral operation is in progress. The station response continues to be no traffic to general polls. A THRU (DLE ;) status is sent as a poll response upon completion of the peripheral operation.

- **UTS 400 Rule 12:**

  Upon completion of a data transfer to or from a peripheral, the UTS 400 sends the THRU (DLE ;) status as a poll response. When automatic peripheral retry is enabled in a station, the THRU response means the peripheral operation was completed successfully. Without automatic peripheral retry, the data transfer may have been completed successfully, but the operation of certain devices (specifically, the tape cassette system) may not be complete when the THRU response is returned.

- **Host Processor Rule 13:**

  If a no-response condition exists on a selection poll, then host processor rule 6 (which requires sending of a specific poll) must be followed. If the response to this specific poll is a reply request, then a retransmission request must be sent. If the response is no traffic, the selection poll must be resent.

- **Host Processor Rule 14:**

  A selection poll can contain an acknowledgment only if the selection is being performed on the station that is owed the acknowledgment.

→ **Host Processor Rule 15:**

  The host processor must maintain peripheral-operation timers at the station level and/or at the UTS 400 cluster level to provide an indication of excessive wait time or of a sustained busy condition. These timers must take into consideration the amount of time a station is on the PAQ prior to performance of the operation (that is, the wait time) and the amount of time required to actually perform the operation. Wait time depends on the number of stations in the cluster, the number of host-processor- and operator-initiated operations already in the PAQ, the type of operations being performed, and whether or not automatic peripheral retry is in effect. The amount of time required to actually perform the operation depends on the type of operations being performed and whether or not automatic peripheral retry is in effect.

  The station-level timer should be initiated for:

  – A delayed status response to a selection poll

  – An ACK plus delayed status response to the poll following a TEXT/SD message

  – An ACK plus message-queued response to the poll following a TEXT/SD/PI message

  – A WABT response to the poll following a TEXT/PI or TEXT/SD/PI message

  When general polls are being used, the station-level peripheral timeouts can be used as a timing mechanism for periodically sending specific polls to determine when the peripheral operation is actually in progress (see UTS 400 rule 11).

  The station-level timer should be terminated upon notification of a peripheral completion (the THRU response if the first response was WABT or ACK plus message queued, or the peripheral device status if the first response was delayed status).

  A cluster-level timer should be initiated for the same poll responses as those listed for a station-level timer. However, once a cluster-level timer is started, similar poll responses from the cluster will not be used to restart the timer. Upon notification of a peripheral completion (THRU if the first response from the station was WABT or ACK plus message queued, or peripheral device status if the first response was delayed status), the host processor must reinitiate the cluster-level timer if there are other host-initiated operations pending for that cluster.

3.8.5. Example of Normal Selection Poll Message Sequence

The following message sequence applies UTS 400 rule 8.

In this sequence, it is assumed that the peripheral is a COP with a DID of s and that the station in the UTS 400 cluster has a RID of 1 and a SID of b (as illustrated in Figure 3–1).

1.	**SOH 1 b s ETX BCC**	⟶	
2.		⟵	**SOH 1 b s DLE > ETX BCC**
3.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
4.		⟵	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. Selection poll.
2. Peripheral ready status (DLE >). (See UTS 400 rule 8.)
3. General traffic poll with acknowledgment.
4. No traffic.

3.8.6 Examples of Line Errors During Selection Poll Message Sequences

Example Sequence 1:

The following message sequence applies UTS 400 rule 8 and host processor rule 13.

In this sequence, it is assumed that a COP with an s DID is to be selected for the station whose RID is 1 and SID is a.

1.	**SOH 1 a s ETX BCC**	⟶	
2.		No-response condition	
3.	**SOH 1 a p ETX BCC**	⟶	
4.		⟵	**EOT EOT ETX BCC**
5.	**SOH 1 a s ETX BCC**	⟶	
6.		⟵	**SOH 1 a s DLE < ETX BCC**
7.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
8.		⟵	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. Selection poll.
2. No-response condition indicates transmission error occurred on poll or response.
3. Specific traffic poll. (See host processor rule 13.)
4. No traffic.
5. The selection poll sent in step 1 is repeated. (See host processor rule 13.)
6. Peripheral status 2 (DLE <). (See UTS 400 rule 8.)
7. General traffic poll with acknowledgment.
8. No traffic.

Example Sequence 2:

The following message sequence applies UTS 400 rules 5 and 8 and host processor rules 8 and 13.

It is assumed that a COP with an s DID is to be selected for the station whose RID is 1 and SID is a.

1.	**SOH 1 a s ETX BCC**	$\longrightarrow$	
2.		No-response condition	
3.	**SOH 1 a p ETX BCC**	$\longrightarrow$	
4.		$\longleftarrow$	**SOH 1 a s DLE ENQ ETX BCC**
5.	**SOH 1 a p DLE NAK ETX BCC**	$\longrightarrow$	
6.		$\longleftarrow$	**SOH 1 a s DLE < ETX BCC**
7.	**SOH 1 P p DLE 1 ETX BCC**	$\longrightarrow$	
8.		$\longleftarrow$	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. Selection poll.
2. No-response condition indicates transmission error occurred on poll or response.
3. Specific traffic poll. (See host processor rule 8.)

4. Reply request. (See UTS 400 rule 5.)

5. Retransmission request. (See host processor rule 13.)

6. Peripheral status 2 (DLE <). (See UTS 400 rule 8.)

7. General traffic poll with acknowledgment.

8. No traffic.

3.8.7. Example of TEXT/SD Message Sequence

The following message sequence applies UTS 400 rule 8.

In this sequence, it is assumed that the peripheral is a COP with a DID of s and that the UTS 400 station for which the peripheral selection and peripheral operation will occur has a RID of 1 and SID of a (as illustrated in Figure 3–1).

1.	**SOH 1 a s STX data ETX BCC**	⟶	
2.	**SOH 1 P p ETX BCC**	⟶	
3.		⟵	**SOH 1 a s DLE 1 DLE> ETX BCC**
4.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
5.		⟵	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. The TEXT/SD message (that is, a text message containing a specific DID) is directed to the screen of the station.

2. General traffic poll.

3. The ACK (DLE 1) in the poll response indicates that the TEXT/SD message in step 1 was received without error by the station. The selection attempt to the COP resulted in the peripheral-ready (DLE >) status. (See UTS 400 rule 8.) No data was sent across the peripheral interface because the TEXT/SD message does not contain a peripheral initiation command.

4. General traffic poll with acknowledgment.

5. No traffic.

3.8.8. Example of TEXT/SD/PI Message Sequence

The following message sequence applies UTS 400 rules 7, 8, and 12 and host processor rule 15.

In this sequence, it is assumed that the peripheral is a COP with a DID of s and that the UTS 400 station for which the peripheral selection and peripheral operation will occur has a RID of 1 and SID of a (as illustrated in Figure 3–1). The peripheral initiation command to be sent is print, which requires inclusion of the DC2 control character in the host processor text message.

1.	**SOH 1 a s STX data DC2 ETX BCC**	⟶	
2.	**SOH 1 P p ETX BCC**	⟶	
3.		⟵	**SOH 1 a s DLE ? ETX BCC**
4.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
5.		⟵	**EOT EOT ETX BCC**
6.	**SOH 1 P p ETX BCC**	⟶	
7.		⟵	**EOT EOT ETX BCC**
		.	
8.		.	
		.	
9.	**SOH 1 P p ETX BCC**	⟶	
10.		⟵	**SOH 1 a s DLE ; ETX BCC**
11.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
12.		⟵	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. The TEXT/SD/PI message (that is, a text message containing a specific DID and a peripheral initiation command) is directed to the COP.

2. General traffic poll.

3. The WABT (DLE ?) in the poll response indicates a successful peripheral selection, acknowledgment by the station for receiving error-free text, and notification that the peripheral operation was in progress when message 2 (the poll) was received. The station-level timer at the host processor is started for this peripheral operation. (See UTS 400 rules 7 and 8 and host processor rule 15.)

4. General traffic poll with acknowledgment.

5. No traffic. The peripheral operation is still in progress.

6, 7, 8. General traffic polls continue to solicit no-traffic responses. The time allowed by the host processor for the peripheral operation to be completed has not elapsed.

9. General traffic poll.

10. The THRU (DLE ;) in the poll response indicates that the peripheral operation has been successfully completed. (See UTS 400 rule 12.)

11. General traffic poll with acknowledgment.

12. No traffic.

3.8.9. Example of Peripheral Error on TEXT/SD/PI Message Sequence

The following message sequence applies UTS 400 rule 8.

In this sequence, it is assumed that the peripheral is a COP with a DID of s and that the UTS 400 station for which the peripheral selection and peripheral operation will occur has a RID of 1 and SID of a, as illustrated in Figure 3–1. The peripheral initiation command to be sent is print, which requires inclusion of the DC2 control character in the host processor text message.

```
1.  SOH 1 a s STX data DC2 ETX BCC  ------>

2.  SOH 1 P p ETX BCC               ------>

3.                                  <------  SOH 1 a s DLE 1 DLE = ETX BCC

4.  SOH 1 P p DLE 1 ETX BCC         ------>

5.                                  <------  EOT EOT ETX BCC
```

Each message in the preceding sequence will now be explained.

1. The TEXT/SD/PI message is directed to the COP.

2. General traffic poll.

3. The ACK (DLE 1) in the poll response indicates that the TEXT/SD/PI message in step 1 was received without error by the station. The selection attempt to the COP resulted in the no-response (DLE =) peripheral status. (See UTS 400 rule 8.)

4. General traffic poll with acknowledgment.

5. No traffic.

3.10. Example of Text and TEXT/SD/PI Message During Offline Peripheral Output Buffering Operation

The following message sequence applies UTS 400 rules 8 and 9.

The UTS 400 can transmit text to the host processor from the screen or receive text from the host processor to the screen during an offline peripheral output buffering operation. If a TEXT/SD/PI message is received by the UTS 400 under these conditions, it is not acknowledged.

In the following message sequence, it is assumed that the peripheral is a COP with a DID of s and that the UTS 400 station involved has a RID of 1 and a SID of a (as illustrated in Figure 3–1). It is also assumed that an offline peripheral buffering operation is in process.

1.	SOH 1 a p STX data ETX BCC	→	
2.	SOH 1 P p ETX BCC	→	
3.		←	SOH 1 a p DLE 1 ETX BCC
4.	SOH 1 P p DLE 1 ETX BCC	→	
5.		←	EOT EOT ETX BCC
6.	SOH 1 a s STX data DC2 ETX BCC	→	
7.	SOH 1 P p ETX BCC	→	
8.		No-response condition	
9.	SOH 1 P p ETX BCC	→	
10.		←	EOT EOT ETX BCC
11.	SOH 1 a s STX data DC2 ETX BCC	→	
12.	SOH 1 P p ETX BCC	→	
13.		←	SOH 1 a s DLE ? ETX BCC
14.	SOH 1 P p DLE 1 ETX BCC	→	
15.		←	EOT EOT ETX BCC
		.	
16.		.	
		.	
17.	SOH 1 P p ETX BCC	→	
18.		←	SOH 1 a s DLE ; ETX BCC
19.	SOH 1 P p DLE 1 ETX BCC	→	
20.		←	EOT EOT ETX BCC

Each message in the preceding sequence will now be explained.

1. Host processor text to the station whose RID is 1 and SID is a.

2. General traffic poll.

3. Acknowledgment from the station that was sent text in step 1. This text is displayed on the screen and this acknowledgment is sent to the host processor even though an offline peripheral output buffering operation is in process.

4. General traffic poll with acknowledgment.

5. No traffic.

6. The TEXT/SD/PI message (a text message containing a specific DID and a peripheral initiation command) is directed to the COP.

7. General traffic poll.

8. No-response condition. The UTS 400 does not respond because of the offline peripheral output buffering operation. This condition appears to the host processor as if an error had occurred on the poll or response.

9. General traffic poll is repeated.

10. No traffic.

11. The TEXT/SD/PI message originally sent in step 6 is repeated and again directed to the COP.

12. General traffic poll.

13. The response to the poll is the WABT (DLE ?) message, which indicates that the offline peripheral output buffering operation is completed, the TEXT/SD/PI message is accepted, and message processing has begun.

14,
15, General traffic polls continue to solicit no-traffic responses.
16.

17. General traffic poll.

18. The THRU (DLE ;) in the poll response indicates that the processing of the TEXT/SD/PI message has been completed successfully.

19. General traffic poll with acknowledgment.

20. No traffic.

3.8.11. Example of Message-Queued Message Sequence

The following message sequence applies UTS 400 rules 7, 8, 11, and 12 and host processor rule 15.

In this sequence, it is assumed that the peripherals are a COP whose DID is s and an 800 terminal printer whose DID is t, that the UTS 400 master station has a RID of 1 and a SID of a, and that a UTS 400 slave in the same cluster has a RID of 1 and SID of b, as illustrated in Figure 3–1. The peripheral initiation command to be sent is print, which requires inclusion of the DC2 control character in the host processor text message.

1.	SOH 1 a s STX data DC2 ETX BCC	⟶	
2.	SOH 1 P p ETX BCC	⟶	
3.		⟵	SOH 1 a s DLE ? ETX BCC
4.	SOH 1 P p DLE 1 ETX BCC	⟶	
5.		⟵	EOT EOT ETX BCC
6.	SOH 1 b t STX data DC2 ETX BCC	⟶	
7.	SOH 1 P p ETX BCC	⟶	
8.		⟵	SOH 1 b t DLE 1 DLE 4 ETX BCC
9.	SOH 1 P p DLE 1 ETX BCC	⟶	
10.		⟵	SOH 1 a s DLE ; ETX BCC
11.	SOH 1 P p DLE 1 ETX BCC	⟶	
12.		⟵	EOT EOT ETX BCC
13.	SOH 1 P p ETX BCC	⟶	
14.		⟵	EOT EOT ETX BCC
15.	SOH 1 P p ETX BCC	⟶	
16.		⟵	SOH 1 b t DLE ; ETX BCC
17.	SOH 1 P p DLE 1 ETX BCC	⟶	
18.		⟵	EOT EOT ETX BCC

Each message in the preceding sequence will now be explained.

1. The TEXT/SD/PI message is directed to the COP. The data is placed in the CRT refresh memory of the UTS 400 master whose RID is 1 and SID is a and appears on the screen of that station.

2. General traffic poll.

3. The WABT (DLE ?) in the poll response indicates a successful peripheral selection, an acknowledgment by the station for receiving error-free text, and notification that the peripheral operation was in progress when message 2 (the poll) was received. The station-level timer at the host processor is started for this peripheral operation. (See UTS 400 rules 7 and 8 and host processor rule 15.)

4. General traffic poll with acknowledgment.

5. No traffic. (The time allowed by the host processor for the peripheral operation to be completed has not elapsed.)

6. The TEXT/SD/PI message is directed to the 800 terminal printer. The data is placed in the CRT refresh memory of the UTS 400 slave whose RID is 1 and SID is b and appears on the screen of that station, even though the message is queued awaiting access to the peripheral interface.

7. General traffic poll.

8. ACK plus message-queued station status. The peripheral interface is busy servicing the UTS 400 master whose RID is 1 and SID is a. The station-level timer at the host processor is started for this peripheral operation. (See UTS 400 rules 8 and 11 and host processor rule 15.)

9. General traffic poll with acknowledgment.

10. The peripheral operation for the master station has been completed successfully. (See UTS 400 rule 12.)

11. General traffic poll with acknowledgment.

12. No traffic.

13. General traffic poll.

14. No traffic.

15. General traffic poll.

16. The peripheral operation for the slave station has been completed successfully.

17. General traffic poll with acknowledgment.

18. No traffic.

3.8.12. Example of Peripheral-Selection-Delayed Message Sequence

The following message sequence applies UTS 400 rules 8 and 10.

This sequence is based on the master/slave cluster configuration illustrated in Figure 3–1. It is assumed that the peripheral being used by the station operator is the tape cassette system and the peripheral being accessed by the host processor is a COP whose DID is s. The station operator is at the UTS 400 master. A slave in the cluster has a RID of 1 and SID of b.

1.	The master station operator presses the PRINT key		
2.	**SOH 1 b s ETX BCC**	$\longrightarrow$	
3.		$\longleftarrow$	**SOH 1 b s DLE 5 ETX BCC**
4.	**SOH 1 P p DLE 1 ETX BCC**	$\longrightarrow$	
5.		$\longleftarrow$	**EOT EOT ETX BCC**
6.	**SOH 1 P p ETX BCC**	$\longrightarrow$	
7.		$\longleftarrow$	**EOT EOT ETX BCC**
8.	**SOH 1 P p ETX BCC**	$\longrightarrow$	
9.		$\longleftarrow$	**SOH 1 b s DLE > ETX BCC**
10.	**SOH 1 P p DLE 1 ETX BCC**	$\longrightarrow$	
11.		$\longleftarrow$	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. A print operation is initiated by the station operator.
2. Selection poll. The attempted selection of the COP is for the slave station whose RID is 1 and SID is b.
3. The peripheral selection attempt is delayed because the peripheral interface is already in use. The station that received the selection poll (step 2) is put in the PAQ. (See UTS 400 rule 10.)
4. General poll with acknowledgment.
5. No traffic.
6. General traffic poll.
7. No traffic. The station that received the selection poll is still not at the top of the PAQ.
8. General traffic poll.

9. The station that received the selection poll has reached the top of the PAQ, and the peripheral-ready status is returned. (See UTS 400 rule 11.)

10. General traffic poll with acknowledgment.

11. No traffic.

3.8.13. Example of Sustained-Busy Condition in a Message Sequence

The following message sequence applies UTS 400 rules 7, 8, and 10 and host processor rule 15.

In this sequence, it is assumed that the peripheral is a COP whose DID is s and that the station in the UTS 400 cluster to which the messages are directed has a RID of 1 and SID of b.

1.	**SOH 1 b s STX data DC2 ETX BCC**	⟶	
2.	**SOH 1 P p ETC BCC**	⟶	
3.		⟵	**SOH 1 b s DLE ? ETX BCC**
4.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
5.		⟵	**EOT EOT ETX BCC**
6.	**SOH 1 P p ETX BCC**	⟶	
7.		⟵	**EOT EOT ETX BCC**
		.	
		.	
		.	
8.	Peripheral operation timeout		
9.	**SOH 1 b p ETX BCC**	⟶	
10.		⟵	**SOH 1 b s DLE ? ETX BCC**
11.	**SOH 1 b s DLE 1 ETX BCC**	⟶	
12.		⟵	**SOH 1 b s DLE < ETX BCC**
13.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
14.		⟵	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. The TEXT/SD/PI message is directed to the COP.
2. General traffic poll.
3. The WABT (DLE ?) in the poll response indicates a successful peripheral selection, acknowledgment by the station for receiving error-free text, and notification that the peripheral operation was in progress when message 2 (the poll) was received. The station-level timer at the host processor is started for this peripheral operation. (See UTS 400 rule 11 and host processor rule 15.)
4. General traffic poll with acknowledgment.
5. No traffic. The time allowed by the host processor for the peripheral operation to be completed has not elapsed.
6. General traffic poll.
7. No traffic.
8. The peripheral operation timeout value is the time allowed by the host processor for a peripheral operation to be completed. (See host processor rule 15.)
9. The host processor has determined a peripheral timeout and sends a specific poll.
10. The response to the specific poll is the WABT (DLE ?) message, which indicates a sustained-busy condition. (See UTS 400 rule 11 and host processor rule 15.)
11. Selection poll with acknowledgment. (See host processor rule 14.)
12. The response to the selection poll is peripheral status 2 (DLE <). The COP may be out of paper. (See UTS 400 rule 8.)
13. General traffic poll with acknowledgment.
14. No traffic.

3.8.14. Example of Message-Queued Sequence With Potential Sustained-Busy Condition

The following message sequence applies UTS 400 rules 7, 8, 10, and 11 and host processor rule 15.

In this sequence, it is assumed that the peripheral is a COP whose DID is s. A station in the UTS 400 cluster has a RID of 1 and SID of b. Operators of stations other than the one with the SID of b in the UTS 400 cluster are accessing peripherals by use of the PRINT key and the XFER key.

```
1.  SOH 1 b s STX data DC2 ETX BCC  ──→

2.  SOH 1 P p ETX BCC               ──→

3.                                  ←── SOH 1 b s DLE 1 DLE 4 ETX BCC
```

4.	SOH 1 P p DLE 1 ETX BCC	⟶	
5.		⟵	EOT EOT ETX BCC
6.	SOH 1 P p ETX BCC	⟶	
7.		⟵	EOT EOT ETX BCC
		.	
		.	
		.	
8.	Peripheral operation timeout		
9.	SOH 1 b p ETX BCC	⟶	
10.		⟵	SOH 1 b s DLE 4 ETX BCC
11.	SOH 1 P p DLE 1 ETX BCC	⟶	
12.		⟵	EOT EOT ETX BCC
13.	SOH 1 P p ETX BCC	⟶	
14.		⟵	EOT EOT ETX BCC
		.	
		.	
		.	
15.	Peripheral operation timeout		
16.	SOH 1 b p ETX BCC	⟶	
17.		⟵	SOH 1 b s DLE ? ETX BCC
18.	SOH 1 P p DLE 1 ETX BCC	⟶	
19.		⟵	EOT EOT ETX BCC
20.	SOH 1 P p ETX BCC	⟶	
21.		⟵	SOH 1 b s DLE ; ETX BCC
22.	SOH 1 P p DLE 1 ETX BCC	⟶	
23.		⟵	EOT EOT ETX BCC

Each message in the preceding sequence will now be explained.

1. TEXT/SD/PI message directed to the COP.

2. General traffic poll.

3. ACK plus message-queued station status (DLE 1 plus DLE 4). The peripheral interface is in use servicing a station in the UTS 400 cluster other than the one whose RID is 1 and SID is b. The host processor starts the station-level timer for this peripheral operation. (See UTS 400 rules 8 and 11 and host processor rule 15.)

4. General traffic poll with acknowledgment.

5. No traffic.

6. General traffic poll.

7. No traffic.

8. The peripheral operation timeout is used by the host processor as a timing mechanism to periodically send specific polls to the station whose RID is 1 and SID is b. This station did not have access to the peripheral interface when it first received the TEXT/SD/PI message. Thus, the response to the periodic specific polls will be message queued (DLE 4) until the message reaches the top of the PAQ. (See UTS 400 rule 11.)

9. Specific traffic poll.

10. The response to the specific traffic poll is message queued (DLE 4). The host processor restarts the station-level timer for the peripheral operation (which has not yet been initiated) and continues sending general traffic polls.

11. General traffic poll with acknowledgment.

12. No traffic.

13. General traffic poll.

14. No traffic.

15. Peripheral operation timeout for the second time. A specific traffic poll must be sent as was done in step 9.

16. Specific traffic poll.

17. The response to the specific traffic poll is WABT (DLE ?). The host processor restarts the station-level timer for the peripheral operation (which has now been initiated) for the last time.

18. General traffic poll with acknowledgment.

19. No traffic.

20. General traffic poll.

21. The THRU (DLE ;) response indicates successful completion of the peripheral operation initiated by the TEXT/SD/PI message in step 1. (See UTS 400 rule 12.)

22. General traffic poll with acknowledgment.

23. No traffic.

3.8.15. Example Message Sequence Involving Passing of the ACK and WABT

The following message sequence applies the multiplexer function definitions (3.6) and host processor rules 3, 4, and 5.

In this sequence, it is assumed that the poll group defined by the UTS 400 master/slave cluster shown in Figure 3–1 is being used. The peripherals are a COP whose DID is s and an 800 terminal printer whose DID is t. Host processor text is directed to the COP through the UTS 400 master whose RID is 1 and SID is a. Host processor text is directed to the 800 terminal printer through the UTS 400 slave whose RID is 1 and SID is c. The peripheral initiation command to be sent is print, which requires inclusion of the DC2 control character in the host processor text message. Host processor text is sent to the screen only of the UTS 400 slave whose RID is 1 and SID is b.

```
 1.  SOH 1 a s STX data DC2 ETX BCC  ──────►
 2.  SOH 1 P p ETX BCC               ──────►
 3.                                  ◄──────  SOH 1 b p DLE ? STX data ETX BCC
 4.  SOH 1 c t STX data DC2 ETX BCC  ──────►
 5.  SOH 1 P p DLE 1 ETX BCC         ──────►
 6.                                  ◄──────  SOH 1 a s DLE ? DLE ; ETX BCC
 7.  SOH 1 b p STX data ETX BCC      ──────►
 8.  SOH 1 P p DLE 1 ETX BCC         ──────►
 9.                                  ◄──────  SOH 1 c t DLE 1 DLE ; ETX BCC
10.  SOH 1 P p DLE 1 ETX BCC         ──────►
11.                                  ◄──────  EOT EOT ETX BCC
```

Each message in the preceding sequence will now be discussed.

1. The TEXT/SD/PI message is directed to the COP. The data is placed in the CRT refresh memory for the UTS 400 master whose RID is 1 and SID is a.

2. General traffic poll.

3. Acknowledgment (busy) plus traffic where the traffic is text. The text is from the station whose RID is 1 and SID is b. The acknowledgment was passed by the UTS 400 master from the station that was sent text in step 1. (See host processor rule 3.)

4. The TEXT/SD/PI message is directed to the 800 terminal printer. The data is placed in the CRT refresh memory for the UTS 400 slave whose RID is 1 and SID is c. (See host processor rule 5.)

5. General traffic poll with acknowledgment.

6. Acknowledgment (busy) plus traffic where the traffic is a THRU response. The THRU is from the station whose RID is 1 and SID is a (the one sent text in step 1). The acknowledgment was passed by the UTS 400 master from the station that was sent text in step 4. (See host processor rules 3 and 4.)

7. The host processor text message is directed only to the screen of the station whose RID is 1 and SID is b.

8. General traffic poll with acknowledgment.

9. Acknowledgment (ACK) plus traffic where the traffic is a THRU response. The THRU is from the station whose RID is 1 and SID is c (the one sent text in step 4). The acknowledgment was passed by the UTS 400 master from the station that was sent text in step 7. (See host processor rules 3 and 4.)

10. General traffic poll with acknowledgment.

11. No traffic.

With a change in RID value, the preceding example is equally valid for the poll group defined by the UTS 400 controller/slave cluster or for the poll group defined by the three UTS 400 masters connected to a terminal multiplexer, as shown in Figure 3–1.

3.8.16. Example of Discarding Duplicate Poll (Retransmission Request) Response

The following message sequence applies the multiplexer function definitions (3.6), UTS 400 rules 5 and 6, and host processor rules 3, 5, 8, 9, and 12.

In this sequence, it is assumed that the poll group defined by the UTS 400 master/slave cluster shown in Figure 3–1 is being used. The peripheral is a COP whose DID is s. The peripheral initiation command to be sent is print, which requires inclusion of the DC2 control character in the host processor text message.

NOTE:

Application of host processor rule 12 to the following example is only one of many possible message sequences to which this rule applies.

```
1.  SOH 1 P p ETX BCC                ──────►
2.                                   ◄──────  EOT EOT ETX BCC
3.  SOH 1 b s STX data DC2 ETX BCC   ──────►
4.  SOH 1 P p ETX BCC                ──────►
5.                                   ◄──────  SOH 1 b s DLE ? ETX BCC
```

6.	SOH 1 P p DLE 1 ETX BCC	$\longrightarrow$	
7.		$\longleftarrow$	EOT EOT ETX BCC
8.	SOH 1 a p STX data ETX BCC	$\longrightarrow$	
9.	SOH 1 P p ETX BCC	$\longrightarrow$	
10.		$\longleftarrow$	SOH 1 b s DLE 1 DLE ; ETX BCC
11.	SOH 1 c p STX data ETX BCC	$\longrightarrow$	
12.	SOH 1 P p DLE 1 ETX BCC	$\longrightarrow$	
13.		No-response condition	
14.	SOH 1 P p ETX BCC	$\longrightarrow$	
15.		$\longleftarrow$	SOH 1 b s DLE ENQ ETX BCC
16.	SOH 1 b p DLE NAK ETX BCC	$\longrightarrow$	
17.		$\longleftarrow$	SOH 1 b s DLE 1 DLE ; ETX BCC
18.	SOH 1 P p DLE 1 ETX BCC	$\longrightarrow$	
19.		$\longleftarrow$	SOH 1 b s DLE 1 ETX BCC
20.	SOH 1 P p DLE 1 ETX BCC	$\longrightarrow$	
21.		$\longleftarrow$	EOT EOT ETX BCC

Each message in the preceding sequence will now be explained.

1. General traffic poll.

2. No traffic.

3. The TEXT/SD/PI message is directed to the COP. The data is placed in the CRT refresh memory for the UTS 400 slave whose RID is 1 and SID is b.

4. General traffic poll.

5. Acknowledgment (busy) from the station that was sent text in step 3.

6. General traffic poll with acknowledgment.

7. No traffic.

8. Host processor text to the station whose RID is 1 and SID is a. (See host processor rule 5.)

9. General traffic poll.

10. Acknowledgment (ACK) plus traffic where the traffic is a THRU response. The acknowledgment was passed by the UTS 400 master from the station that was sent text in step 8. (See host processor rule 3.)

11. Host processor text to the station whose RID is 1 and SID is c. (See host processor rule 5.)

12. General traffic poll with acknowledgment.

13. No-response condition indicates transmission error on poll or response.

14. The general traffic poll is repeated without acknowledgment. (See host processor rule 8.)

15. Reply request from the station whose RID is 1 and SID is b. (See UTS 400 rule 5.)

16. Retransmission request (DLE NAK) to the station that sent the reply request in step 15. Host processor rule 9 applies because the UTS 400 message (step 10) sent previous to the reply request did not include text data.

17. This message is a retransmission (duplication) of the message in step 10. Host processor rule 12 applies because the message (step 10) sent previous to the reply request is of the same type (ACK plus THRU) and from the same station (RID is 1 and SID is b) as this message.

18. General traffic poll with acknowledgment.

19. Acknowledgment (ACK) for the host processor text sent in step 11. This ACK was passed by the multiplexer function (from the station whose RID is 1 and SID is c to the station whose RID is 1 and SID is b) at the time the general traffic poll was received by the UTS 400 in step 14. However, this ACK was not sent with the reply request in step 15. (See UTS 400 rule 6.) Also, it was not the ACK sent in step 17.

20. General traffic poll with acknowledgment.

21. No traffic.

3.8.17. Contention

The UTS 400 method of sharing peripherals among stations gives rise to three possible forms of contention:

- Host/host
- Host/operator
- Operator/operator

Only the first two forms are of concern to the host software.

Host/host contention is resolved through the host handling of the message-queued response, the delayed-status response, and the peripheral timeout considerations.

Host/operator contention on different stations is handled by the host software that handles host/host contention. Host/operator contention on the same station can be resolved by several mechanisms. If operator initiations are to take priority, then the host alerts the operator of its requirement for access by way of the message-waiting light (using the BEL sequence). If the host has priority, host processor messages containing DIDs other than 70 (hexadecimal) will terminate an operator-initiated peripheral operation currently using the screen, thus freeing the screen to accommodate processor text.

The host software must resolve the contentions in a manner that allows reliable interleaving between operator and host or between host and host-initiated peripheral operations. Note that, because of the peripheral-sharing aspects of the UTS 400, the responsibility for file and form control is placed on the user.

3.9. UTS 400 MESSAGE PRIORITIES

3.9.1. Rules for UTS 400 Message-Sending Priority

The following rules define the priorities in which messages are transmitted when a UTS 400 station or cluster has more than one message to send.

- **UTS 400 Rule 13:**

  A station will not respond with a message-waiting or program-attention key code message to a poll containing an acknowledgment if that station is owed an acknowledgment.

- **UTS 400 Rule 14:**

  If a UTS 400 station or cluster has more than one response to send, the following priority sequence will be observed:

  1. Any station or peripheral status responses are sent first. These responses include:

     a. Message-queued response (DLE 4) due to receipt of a TEXT/PI or TEXT/SD/PI message when other messages were already in the PAQ

     b. Peripheral-selection-delayed response (DLE 5) due to receipt of a selection poll or a TEXT/SD message when other messages were already in the PAQ

     c. Peripheral-operation-THRU response (DLE ;) due to completion of the peripheral operation

     d. Peripheral status response 1 (DLE >) due to receipt of a TEXT/SD message or a selection poll when no other messages were in the PAQ

     e. Peripheral status response 2 (DLE <), 3 (DLE :) or 4 (DLE =) due to receipt of a selection poll, a TEXT/SD message, or a TEXT/SD/PI message when no other messages were in the PAQ

  2. If no priority 1 conditions exist, then any text messages resulting from the XMIT key being pressed or from a host processor transmit command are sent.

  3. If no priority 2 conditions exist, then any message-waiting or program-attention key code messages are sent.

3.9.2. Example Sequence Where Program Attention Key Is Pressed During Host Processor Text Message

The following message sequence applies UTS 400 rule 14.

In this sequence, it is assumed that the station has a RID of 1 and SID of a.

1.	**SOH 1 a p STX data ETX BCC**	⟶	
2.	Program attention key F1 is pressed while the text message in step 1 is arriving at the station.		
3.	**SOH 1 P p ETX BCC**	⟶	
4.		⟵	**SOH 1 a p DLE 1 7 ETX BCC**
5.	**SOH 1 P p DLE 1 ETX BCC**	⟶	
6.		⟵	**EOT EOT ETX BCC**

Each message in the preceding sequence will now be explained.

1. Text is sent to the screen of the station whose RID is 1 and SID is a.
2. The function keys are not locked even though data is arriving at the station.
3. General traffic poll.
4. The ACK (DLE 1) response indicates that the text message in step 1 was received without error. The 7 indicates that program attention key F1 was pressed before the poll sent in step 3 had arrived. The station had no higher-priority messages to send. (See UTS 400 rule 14.)
5. General traffic poll with acknowledgment.
6. No traffic.

3.10. DISCONNECTION OF THE COMMUNICATIONS LINE

A UTS 400 feature is available that provides for automatic disconnection of the communications line. This feature enables the UTS 400 to alert the host processor that the UTS 400 end of the line is going to be disconnected and, therefore, that the host processor end of the line should also be disconnected. The host processor has an equivalent capability if this feature is present in the UTS 400.

3.10.1. Disconnection at the UTS 400

The UTS 400 responds to a disconnection message from the host (SOH RID SID DID DLE EOT STX ETX BCC) with an acknowledgment (DLE 1) to the next poll that station recognizes. When the host processor acknowledges that response, the UTS 400 disconnects from the communications line.

All host processor polls are timed by the UTS 400 if the disconnection feature is present. Any timeout between polls exceeding a predetermined value causes the UTS 400 to disconnect from the communications line. Selectable timeout values are as follows: 25.5 seconds, 2.75 minutes, 8.25 minutes, and 11 minutes. The selected value may be factory installed or changed in the field by a Sperry Univac customer engineer.

The timeout/disconnection sequence is also enabled by pressing the HANG UP key on the keyboard or by a ring indication from the terminal modem.

3.10.2. Disconnection at the Host Processor

Upon receipt of a disconnection message from the UTS 400 (SOH RID SID DID DLE EOT ETX BCC), the host processor must acknowledge the message before disconnecting from the communications line.

3.11. HOST PROCESSOR PROGRAMMING SUGGESTIONS

3.11.1. Handling of ACK/WABT-Plus-Traffic Responses

In a poll group, the UTS 400 multiplexer function can combine an acknowledgment from one station with any traffic response from another station. This process of combining responses is called "passing the ACK/WABT response."

When an ACK/WABT-plus-traffic response is returned by a station that just received host processor text, the ACK/WABT portion of the response is from that station and was not passed from another station by the multiplexer function. Possible ACK/WABT-plus-traffic poll responses from a station that just received host processor text are as follows:

- ACK plus message waiting key code (refer to 3.9.2)
- ACK plus program attention key code (refer to 3.9.2)
- ACK plus peripheral selection delayed (refer to UTS 400 rule 8)
- ACK plus message queued (refer to UTS 400 rule 8)
- ACK plus peripheral status (refer to UTS 400 rule 8)
- ACK plus text (poll response following host processor text containing a transmit function)
- ACK plus THRU (refer to UTS 400 rule 7)
- WABT plus message waiting key code (refer to 3.9.2)
- WABT plus program attention key code (refer to 3.9.2)

Passing of the ACK/WABT response by the multiplexer function does not take place, and all considerations concerning the passing of this response are eliminated, under the following conditions:

- When host processor text to a station is followed by a specific poll to that station, or
- When each poll group has only one station and, therefore, a general poll is recognized only by that station.

However, the use of general polls and the configuration of poll groups with more than one station produce the most efficient communications exchange (that is, the minimum number of messages).

The host processor may be programmed to handle ACK/WABT-plus-traffic responses resulting from general polls by retaining, at the line level, the address of the station from which the host processor expects an acknowledgment. If the station address contained in the response is that of the station from which an acknowledgment is expected, then both the ACK/WABT and the traffic portion of the response would be processed for that station. If, however, the station address contained in the response is different from the address of the station from which an acknowledgment is expected, then the ACK/WABT portion of the response would be processed for the station from which an acknowledgment is expected, and the traffic portion would be processed for the station whose address is in the response.

3.11.2. Passing the ACK/WABT Response in TEXT/SD, TEXT/PI, or TEXT/SD/PI Messages

Whenever TEXT/SD, TEXT/PI, or TEXT/SD/PI messages are being sent by the host processor, the ACK in the response must not be interpreted as meaning a successful peripheral selection and/or successful completion of the peripheral operation. In the ACK-plus-traffic combinations shown in Table 3–7 (peripheral status responses), it is possible for the ACK portion of the response to be passed by the multiplexer function to another station; if this is done, the response to a general poll following the host processor text message will contain only the ACK portion of the original ACK-plus-traffic response. Therefore, the host processor must wait for a subsequent poll response containing the traffic portion of the original ACK-plus-traffic response to determine the status of the peripheral selection and/or operation.

The following message sequence example illustrates how the ACK should be interpreted.

```
1.  SOH 1 a s STX data DC2 ETX BCC  ──────►
2.  SOH 1 P p ETX BCC               ──────►
3.                                  ◄──────  SOH 1 b p DLE 1 STX data ETX BCC
4.  SOH 1 P p DLE 1 ETX BCC         ──────►
5.                                  ◄──────  SOH 1 a s DLE 4 ETX BCC
6.  SOH 1 P p DLE 1 ETX BCC         ──────►
7.                                  ◄──────  EOT EOT ETX BCC
```

Each message in the preceding sequence will now be explained.

1. TEXT/SD/PI message directed to the COP through the station whose RID is 1 and SID is a.
2. General traffic poll to the poll group whose RID is 1.
3. ACK-plus-traffic message where the traffic is text from the station whose RID is 1 and SID is b. The ACK was passed from the station that was sent the TEXT/SD/PI message in step 1.
4. General traffic poll with acknowledgment.
5. Message queued (DLE 4) response. Originally, the station response was to contain ACK plus message queued (DLE 1 DLE 4); however, the ACK from the station was passed by the multiplexer function and had already been sent in step 3.
6. General traffic poll with acknowledgment.
7. No traffic.

3.11.3. Line Error Recovery

The UTS 400 is designed to allow recovery from communications line errors without loss or duplication of messages. To do this, the line error recovery rules discussed in 3.7 must be followed.

The following guidelines should be observed at the host processor to facilitate line error recovery:

- The error recovery sequence should be completed when the line error occurs.
- The address of the station which was just sent host processor text, and therefore owes the host processor an acknowledgment, should be retained.
- Not more than one station at a time should be put on the line in a state where it has an acknowledgment to send.
- The previous poll response type (such as UTS 400 text, ACK, or ACK plus THRU) should be retained.
- The address of the previous poll response should be retained.

4. UTS 400 and Host Processor Intercommunication

4.1. GENERAL INFORMATION

UTS 400 operator and host processor data exchange takes place by way of the UTS 400 display screen. Text characters from the host processor to the UTS 400 are placed on the screen, and text characters from the UTS 400 to the host processor are sent from the screen.

Host processor text is also directed to or from the share-type peripherals by way of the display screen. (If the optional screen bypass feature is present, the host processor can send text to or receive text from the share-type peripherals without displaying the text on the screen. However, this text flow to and from the host processor is structured as if the text were going to and from a screen.) Text to be sent from the UTS 400 to a share-type peripheral is also transferred by way of the display screen.

The screen display will have one of the following formats:

Total Lines (rows)	Character Positions (columns) per Line	Total Character Positions
12	80	960
16	64	1024
24	64	1536
24	80	1920

Characters received from the communications line, from the keyboard, or from a nonshare- or share-type peripheral are entered into UTS 400 display storage and then displayed on the screen. The character is placed on the screen at the position occupied by the cursor, and the cursor is then advanced one position. Each character position is uniquely addressable as explained in 4.2.

The UTS 400 provides the capability for field definition on the screen by means of the field control character (FCC), which gives the host processor or the UTS 400 operator complete control over data formatting. The FCC is similar in concept to a protected-format function, but with many more capabilities. For example, fields may be formed that will accept only numbers or only letters or that will right-justify all entries. The display intensity of the characters in a field may be controlled to aid in visual identification of the data, or all characters in the field may be made to blink. The same FCC used to define the parameters of the field may also be used as a tab stop, thus saving space in the display for additional data. (The FCC occupies a location in memory but not on the display; the regular tab stop requires one space in the screen display.)

Other UTS 400 capabilities available to the host processor, as well as the operator, include the edit functions (erases, deletes, inserts, and line duplication), the cursor control functions (backward tab, forward tab, cursor scans, cursor to home, cursor return, and cursor positioning), and the peripheral control functions, in addition to the basic function of data character entry to the screen.

All control functions and data character sequences sent from the host processor are contained in a text envelope, which is described in 4.2. The text envelope in which characters are sent from the UTS 400 to the host processor (or to a share-type peripheral) is described in 4.3. The control codes and special host processor commands used in the context of text messages to effect UTS 400 and peripheral operations are covered in 4.4 through 4.6.

4.2. HOST PROCESSOR TEXT MESSAGES TO UTS 400

4.2.1. Format

The format for sending text messages from the host processor to a UTS 400 is summarized in Figure 4–1. As shown in the figure, a text message must be bracketed by STX and ETX characters.

$$\text{SOH}\ \underbrace{\text{RID SID DID}}_{\text{Station address}}\ \underbrace{\text{STX data to screen ETX BCC}}_{\text{Text message}}$$

44628

Figure 4–1. Text Message Format to UTS 400 Screen

4.2.2. Cursor Positioning

Text transmitted from the host processor to the UTS 400 is placed on the screen starting at the cursor position. Several different methods may be used by the host processor to position the cursor:

- The cursor address sequence positions the cursor to any specified point on the screen according to row and column coordinates.

- The FCC sequence, covered in 4.4.1.2, also includes codes to designate cursor position according to row and column coordinates.

- The cursor-to-home code places the cursor in the upper left-hand corner of the screen.

- The cursor return and cursor scan codes position the cursor relative to its last position.

- The tabulation control codes move the cursor in a manner similar to the tabulation function of a typewriter (assuming tab stops are present).

These cursor positioning sequences are discussed in the following paragraphs.

4.2.2.1. Cursor Address Sequence

The format for the cursor address in a text message from the host processor to the UTS 400 is shown in Figure 4–2.

$$\text{STX}\ \underbrace{\text{ESC VT Y X SI}}_{\substack{\text{Cursor address sequence}\\ \text{(as required)}}}\ \text{text}\ \underbrace{\text{ESC VT Y X SI}}_{\substack{\text{Cursor address sequence}\\ \text{(as required)}}}\ \text{text}\ \ \text{ETX}$$

44629

Figure 4–2. Cursor Address Sequence With Text

Character definition within the cursor address sequence is as follows:

ESC (escape)
indicates that the following character or characters are part of a control sequence.

VT
indicates that the next two characters are the cursor address.

Y
is the Y coordinate identifying the horizontal line (or row) on the screen where the cursor is to be placed. (Character position assignments are shown in Figure 4–3.)

X
is the X coordinate identifying the vertical column on the screen (Figure 4–3) where the cursor is to be placed.

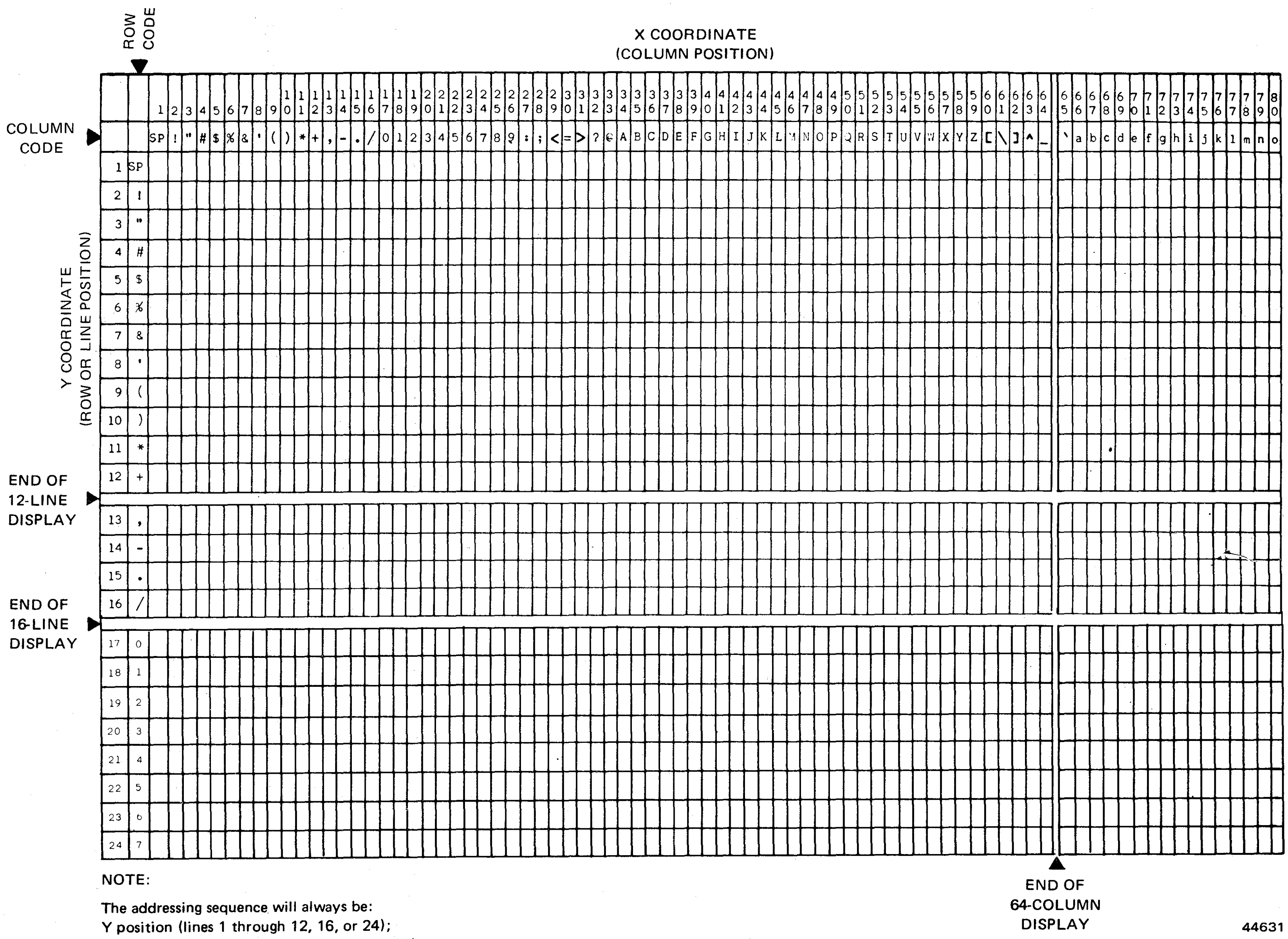
SI
indicates the end of the cursor address sequence.

The cursor address coding shown in Figure 4–3 is derived from columns 2 through 6 of the ASCII code chart (Figure 2–2). Columns 0 and 1 of the ASCII code chart contain control codes and are not used for cursor address codes. Column 7 also is not used because no address greater than 80 is required with the UTS 400. The exact code characters specified are a function of the number of lines and columns in the particular terminal system being addressed and the desired cursor position on the screen.

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.

4.2.2.2. Cursor-to-Home Code (ESC e)

This function causes the cursor to move to the first display position of the first line in the upper left-hand corner of the screen (home position). This position is row 1, column 1 on the screen.


ROW CODE
X COORDINATE (COLUMN POSITION)
COLUMN CODE
SP ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [\ ] ^ _ ` a b c d e f g h i j k l m n o
Y COORDINATE (ROW OR LINE POSITION)
1 SP
2 !
3 "
4 #
5 $
6 %
7 &
8 '
9 (
10)
11 *
12 +
13 ,
14 -
15 .
16 /
17 0
18 1
19 2
20 3
21 4
22 5
23 6
24 7
END OF 12-LINE DISPLAY
END OF 16-LINE DISPLAY
END OF 64-COLUMN DISPLAY
44631

NOTE:

The addressing sequence will always be:
Y position (lines 1 through 12, 16, or 24);
then X position (columns 1 through 64 or 80).

Figure 4–3. Cursor Addressing for Screen Display

4.2.2.3. Cursor Return Code (CR)

This function causes the cursor to advance to the first display position on the next line. If the cursor is on the last line of the screen, it advances to the first display position of the first line (home position). The CR code also acts as a control character for printers. (Refer to 4.6.2.4.)

4.2.2.4. Cursor Scan Codes

- Scan Up (ESC f)

  This function causes the cursor to move up one line in the same column. If the cursor is in the first line at the top of the display, it moves to the same column in the last line of the display.

- Scan Left (ESC g)

  This function causes the cursor to move to the left one character position. If the cursor is in the first character position of a line, it moves to the last character position of the preceding line. If the cursor is in the home position, it moves to the last character position of the last line.

- Scan Right (ESC h)

  This function causes the cursor to move to the right one character position. If the cursor is in the last character position of a line, it advances to the first character position of the following line. If the cursor is in the last character position of the last line, it moves to the first display position of the first line (home position).

- Scan Down (ESC i)

  This function causes the cursor to move down one line in the same column. If the cursor is in the last line of the display, it moves to the same column position in the first line.

4.2.2.5. Forward Tab Code (HT)

Cursor control by means of the forward tab code is discussed in 4.4.6.3.

4.2.2.6. Backward Tab Code (ESC z)

Cursor control by means of the backward tab code is discussed in 4.4.6.4.

4.2.2.7. Cursor Positioning With the FCC Sequence

The FCC sequence includes codes to specify the position of the cursor by row and column. Details of the FCC sequence are covered in 4.4.1.

4.2.3. Code Sequences To Be Avoided in Host Processor Transmissions

Certain ESC sequences, if sent by the host processor to the UTS 400, cause identifiable functions to occur but the results of these functions are not predictable. These ESC sequences are given in Table 4–1.

Table 4–1. Code Sequences To Be Avoided

Sequence	Function	Sequence	Function
ESC J	Recover auxiliary buffer	ESC q	Print
ESC I	Release auxiliary buffer	ESC r	Transfer
ESC m	Back one block	ESC s	FCC generate
ESC n	Search	ESC p	Transmit
ESC v	FCC reenable	ESC L	Keyboard unlock
ESC x	FCC locate	ESC l	Report address
ESC N	Auxiliary status	ESC O	Hang up

4.3. UTS 400 TEXT MESSAGES TO HOST PROCESSOR

4.3.1. Format

Messages from the UTS 400 are sent in response to polls from the host processor. A text message sent from the UTS 400 to the host processor is contained in an envelope bracketed by STX and ETX characters. The character string sent from the display screen begins with the start-of-entry (SOE) character on the screen (or with the home position if no SOE is used) and continues to and including the cursor position. The first control sequence in the text message is the address of the SOE character, as shown in Figure 4–4, or the address of the home position if there is no SOE, as shown in Figure 4–5.

$$\text{STX}\ \underbrace{\text{ESC VT Y X NUL SI}}_{\text{Start-of-entry (SOE) address}}\ \text{RS text ETX BCC}$$

44632

Figure 4–4. Text Message Format, Starting with SOE Character — UTS 400 to Host Processor

$$\text{STX}\ \underbrace{\text{ESC VT SP SP NUL SI}}_{\text{Home position address}}\ \text{text ETX BCC}$$

44633

Figure 4–5. Text Message Format Without SOE Character — UTS 400 to Host Processor

In a text message sent to the host processor, an SOE character located at the beginning of the message is always sent to the host processor with the message, unless the SOE is in an unchanged field and a transmit-change command is sent. Then the SOE is not transmitted. In a text message sent from the UTS 400 to a peripheral device, an SOE character at the beginning of the message is always suppressed.

4.3.2. Transmission Variations

The type of data sent in a message transmitted from the UTS 400 to the host processor is dependent upon the FCC/PROTECT switch setting (4.4.1.4) and the particular transmit mode specified by the operator in the control page definitions. The three possible transmit modes are defined in the following paragraphs. These definitions apply when the FCC/PROTECT switch is in the FCC position. Refer to 4.4.1.4 for an explanation of the functional differences in the transmit modes when the FCC/PROTECT switch is in the PROTECT position.

Regardless of the transmit mode specified, the nonsignificant spaces at the end of a field and at the end of a line (except the line containing the cursor) are suppressed and are not transmitted to the host processor.

4.3.2.1. Transmit All

In the transmit-all-data mode, all fields, both protected and unprotected, and their associated FCCs are transmitted. Data sent is bracketed between the SOE character (or the home position) and the cursor.

4.3.2.2. Transmit Variable

In the transmit-variable-data mode, all unprotected fields and their associated FCCs are transmitted. Data sent is bracketed between the SOE character (or home position) and the cursor.

4.3.2.3. Transmit Changed

In the transmit-changed-data mode, only fields that have been changed, and the FCCs associated with those changed fields, are transmitted. Data sent is bracketed between the SOE character (or the home position) and the cursor.

4.4. SCREEN CONTROL

The UTS 400 provides the host processor and the UTS 400 operator with many types of screen control capabilities. Basic display parameters and field definitions are controlled by means of the FCC. FCC definition and use in text messages are covered in 4.4.1.

Other capabilities include the erase, delete, insert, tabulation, and line duplication functions. These functions and their ASCII control codes are covered in 4.4.2 through 4.4.7.

4.4.1. Field Control Character (FCC) Usage

The discussion in 4.4.1.1 through 4.4.1.3 is based on the assumption that the FCC/PROTECT switch is in the FCC position. Refer to 4.4.1.4 for an explanation of functional differences when the switch is in the PROTECT position.

4.4.1.1. FCC Functions

The following list summarizes the uses of the FCC.

- The FCC designates one of the following display characteristics:
  1. Normal intensity (high)
  2. Low intensity
  3. Display off (characters in the field not displayed)
  4. Blinking display (data in the field alternates between normal and low intensity, thus appearing to blink)
- The FCC specifies one of the following modes of data entry.
  1. Protected (no keyboard data entry allowed)
  2. Unprotected (data entry allowed from keyboard)
     - Any input allowed
     - Alphabetic entry only (uppercase and lowercase alphabetic characters and spaces)
     - Numeric entry only [all numbers and the plus (+), minus (—), comma (,) and decimal (.) symbols]
     - Right justification, any entry (justified to end of field or end of line, whichever comes first)
     - Right justification, alphabetic entry only (justified to end of field or end of line, whichever comes first)
     - Right justification, numeric entry only (justified to end of field or end of line, whichever comes first)
- The FCC specifies its use as a tab stop
- The FCC designates whether a field has or has not been changed by an operator

The FCC occupies a storage location but not a screen location. As many as 15 FCCs can be used in each line of the display. Once entered into screen storage, the FCC remains there until erased by an erase display, insert line, delete line, FCC clear, or line duplication function.

4.4.1.2. FCC Sequence From Host Processor

A specific sequence must be sent by the host processor to generate the FCC function in the UTS 400. This sequence is sent as text, that is, within the STX—ETX envelope. Within this sequence, the cursor position is defined to indicate the beginning of the field. Text following the FCC sequence is sequentially placed, beginning at the defined cursor location, unless there is an intervening cursor-movement code.

The FCC sequence is shown in Figure 4–6.

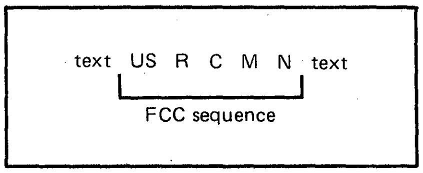


Figure 4–6. FCC Sequence

Characters in the FCC sequence are defined as follows:

US

is the control character used to preface an FCC sequence.

R

represents the number of the row in which the FCC is to be placed.

C

represents the number of the column in which the FCC is to be placed.

M

represents an ASCII character placed in the sequence to define the state of bits 4, 5, 6, and 7 of the FCC. The ASCII characters that may be used are listed in Table 4–2.

N

represents an ASCII character placed in the sequence to define the state of bits 0, 1, 2, and 3 of the FCC. The ASCII characters that may be used are listed in Table 4–3.

4.4.1.3. FCC Sequence From UTS 400

A specific sequence is used by the UTS 400 to transmit an FCC to the host processor. This sequence is sent as text, that is, within the STX–ETX envelope. The FCC sequence is as follows:

US R C M N

Characters in the FCC sequence are defined in 4.4.1.2. Values for M and N are listed in Tables 4–2 and 4–3.

Table 4–2. ASCII Characters Used as M in the FCC Sequence

ASCII Character	Octal Code	Hexadecimal Code	Field Characteristics
0	60	30	Tab stop, normal intensity, changed field*
1	61	31	Tab stop, display off (no intensity), changed field*
2	62	32	Tab stop, low intensity, changed field*
3	63	33	Tab stop, blinking display, changed field*
4	64	34	Tab stop, normal intensity
5	65	35	Tab stop, display off (no intensity)
6	66	36	Tab stop, low intensity
7	67	37	Tab stop, blinking display
8	70	38	Not tab stop, normal intensity, changed field*
9	71	39	Not tab stop, display off (no intensity), changed field*
:	72	3A	Not tab stop, low intensity, changed field*
;	73	3B	Not tab stop, blinking display, changed field*
<	74	3C	Not tab stop, normal intensity
=	75	3D	Not tab stop, display off (no intensity)
>	76	3E	Not tab stop, low intensity
?	77	3F	Not tab stop, blinking display

*Normally, when an FCC is generated by the host processor, the changed-field designator is cleared. However, the host processor can generate individual FCCs with the changed-field designator set; this capability may be used for selective transfer or transmission of fields which were not in fact changed by the UTS 400 operator. By sending an ESC u code (4.5.8) to the UTS 400 in a text message, the host processor can clear the changed-field designators in all FCCs without regenerating each FCC and without altering the data within the fields.

Table 4–3. ASCII Characters Used as N in the FCC Sequence

ASCII Character	Octal Code	Hexadecimal Code	Field Characteristics
0	60	30	Any input allowed
1	61	31	Alpha only allowed
2	62	32	Numeric only allowed
3	63	33	Protected (no entries and no changes allowed)
4	64	34	Any input allowed, right-justified
5	65	35	Alpha only allowed, right-justified
6	66	36	Numeric only allowed, right-justified

4.4.1.4. Implications of FCC/PROTECT Switch Settings

The FCC/PROTECT switch setting affects the following UTS 400 functions:

- Field definition on the UTS 400 display screen by means of the FCC
- Formatting of host processor messages into protected fields (similar to the protected-field concept used with SPERRY UNIVAC UNISCOPE Display Terminals)
- Use of the Katakana character set
- Use of the transmit-mode functions
- Use of program attention keys F5 through F22

UTS 400 recognition of the FCC sequence (US R C M N) in a host processor text message is not affected by the FCC/PROTECT switch setting.

NOTE:

Throughout this manual, UTS 400 functions are described as they occur with the FCC/PROTECT switch in the FCC position.

4.4.1.4.1. Switch in FCC Position

With the FCC/PROTECT switch in the FCC position, the applicable UTS 400 functions are affected as follows:

1. Katakana characters with FCC field definitions may be sent or received by the UTS 400.

   Characters bracketed by the SO and SI or ETX characters in a text message from the host processor are displayed as Katakana characters on the UTS 400 screen.

   Katakana characters displayed on the UTS 400 screen are bracketed by the SO and SI characters when sent from the screen to the host processor.

2. Fields sent to and from the UTS 400 are defined by the FCC sequence, as previously described in 4.4.1.1 through 4.4.1.3.

3. The transmit-mode functions operate as previously described in 4.3.2.

4. Program attention keys F5 through F22 are enabled.

4.4.1.4.2. Switch in PROTECT Position

With the FCC/PROTECT switch in the PROTECT position, the applicable UTS 400 functions are affected as follows:

1. The protected-field definition capability (similar to the UNISCOPE protected-field concept) is in effect. However, internally the UTS 400 defines all fields with the FCC character.

The SO and SI characters are used to define protected fields in the host processor text message (not as Katakana definition characters).

Upon receipt of the SO character, the UTS 400 creates a field at the current cursor location, using an FCC with the following attributes:

– Normal intensity

– Protected entry

– Not tab stop

The next character following the SO will appear at that cursor location.

Upon receipt of the SI character or the ETX character following the SO character in a host processor text message, the UTS 400 terminates the protected field and then defines another field at the current cursor location, using an FCC with the following attributes:

– Normal intensity

– Any entry allowed

– Not tab stop

2. The UTS 400 transmit-mode functions are modified from the definitions given in 4.3.2. The fields defined on the screen are transmitted according to the following transmit conditions:

   – Transmit All

     All characters are transmitted, from the SOE character (or the home position) to and including the cursor, with no distinction between protected and unprotected fields. FCC sequences are not transmitted. Spaces at the end of a line are suppressed except in the line containing the cursor.

   – Transmit Variable

     All unprotected characters are transmitted, from the SOE character (or the home position) to and including the cursor. Protected fields are replaced by one SUB code per field. Cursor return codes are transmitted with every line in both protected and unprotected fields. Nonsignificant spaces at the end of the line are suppressed except in the line containing the cursor. FCC sequences are not transmitted.

   – Transmit Changed

     This function is inoperative when the FCC/PROTECT switch is in the PROTECT position.

3. Program attention keys F5 through F22 are disabled.

NOTE:

When the FCC/PROTECT switch is in the PROTECT position, no Katakana characters can be accepted from or sent to the host processor by the UTS 400.

4.4.2. Erase Functions

4.4.2.1. Erase Unprotected Data (ESC a)

This function erases all unprotected characters from the cursor position to the end of the display, inclusive. FCCs and protected fields are not erased.

4.4.2.2. Erase to End of Field (ESC K)

This function erases all unprotected characters from the cursor position to the end of the field in which the cursor is located or to the end of the display, whichever occurs first. FCCs and protected fields are not erased.

4.4.2.3. Erase to End of Line (ESC b)

This function erases all unprotected characters from the cursor position to the end of the field in which the cursor is located or to the end of the line, inclusive, whichever occurs first. FCCs and protected fields are not erased.

4.4.2.4. Erase Display (ESC M)

This function erases all characters and FCCs from the cursor position to the end of the display, inclusive. Since FCCs are erased, any protected data in the display is also erased.

4.4.2.5. Erase Character (SP)

The space character is used as a blank. When used for the erase-character function, the space character replaces the character in UTS 400 storage located at the cursor position.

4.4.2.6. Clear FCC (ESC w)

This function clears the first field control character to the left of the cursor. The cursor may be located anywhere within the field containing the FCC to be cleared.

4.4.3. Delete Functions

4.4.3.1. Delete in Line (ESC c)

If the cursor is located in an unprotected field, this function causes the character at the cursor position to be deleted. All characters following the cursor position in that field, up to the end of the line, are backspaced one character location. A space appears at the end of the field or at the end of the line, whichever occurs first. The cursor does not move from its original position during this function.

Protected data and FCCs are not affected by the delete-in-line function.

4.4.3.2. Delete in Display (ESC C)

If the cursor is located in an unprotected field, this function causes the character at the cursor position to be deleted. All characters following the cursor position in that field, up to the end of the display, are backspaced one character location. A space appears at the end of the field or at the end of the display, whichever occurs first. The cursor does not move from its original position during this function.

4.4.3.3. Delete Line (ESC k)

This function causes the line in which the cursor is located to be deleted and replaced by the next line down. Any FCCs and protected data in the deleted line are also deleted. All following lines are moved up one line, and a blank line (without FCCs) appears at the bottom of the screen. The cursor does not move from its original position during this function.

4.4.4. Insert Functions

4.4.4.1. Insert Line (ESC j)

This function causes the line in which the cursor is located, and all following lines of the display, to be moved down one line. A blank line (without FCCs) appears at the line occupied by the cursor.

The characters (if any) in the last line on the screen are, in effect, moved down off the screen and are deleted. The cursor does not move from its original position during this function.

4.4.4.2. Insert in Line (ESC d)

If the cursor is located in an unprotected field, this function causes the character at the cursor position, and all following characters to the end of that field or the end of the line, whichever occurs first, to be advanced one character position. A space appears at the cursor position. The character (if any) in the last character position of the field or the line is deleted.

4.4.4.3. Insert in Display (ESC D)

If the cursor is located in an unprotected field, this function causes the character at the cursor position and all following characters to the end of that field or the end of the display, whichever occurs first, to be advanced one character position. A space appears at the cursor position. The character (if any) in the last character position of the field or the display is deleted.

4.4.5. Line Duplication (ESC y)

This function causes the contents of the line in which the cursor is located to be duplicated on the line directly below. The cursor is repositioned to the corresponding character position on the duplicated line. Any data that was originally on the duplicated line is deleted.

Repeating the line duplication function (once per line) allows information to be repeated to the bottom of the screen. In this manner FCCs, tab stops, and text can be duplicated on a column basis.

4.4.6. Tabulation

The UTS 400 has both forward and backward tab capability. In addition to the standard function, which places a nondisplayable tab stop code in a character location on the screen, the UTS 400 also provides a means whereby the FCC may be designated as a tab stop. Using the FCC as a tab stop saves the character location that would be occupied by a tab code.

4.4.6.1. Tab Stop Set (ESC HT)

This function places a tab stop code in display storage at the cursor position and advances the cursor one position. The tab stop code appears as a space on the display screen.

4.4.6.2. FCC Tab Stop

Refer to 4.4.1.2 for the code sequence required to use the FCC as a tab stop.

4.4.6.3. Forward Tab (HT)

This function causes the cursor to move to the first tab stop following the original cursor position. The cursor stops at the first unprotected character position to the right of the tab stop (which may be either an FCC or a tab stop code). If there are no tab stops, or if the entire remaining display is protected, the cursor moves to the home position.

4.4.6.4. Backward Tab (ESC z)

This function initiates a tab operation in the opposite direction from that of a normal (forward) tab operation. The cursor stops at the first unprotected character position to the right of the tab stop (which may be either an FCC or a tab stop code).

4.4.7. Other Functions

4.4.7.1. Space (SP)

This character code is placed in display storage at the cursor position. The space character appears as a blank on the display screen.

4.4.7.2. Blinking Start Marker (FS)

This character code, which is placed in display storage at the cursor position, causes a blinking marker to appear on the display screen to mark the start of a highlighted field.

4.4.7.3. Blinking End Marker (GS)

This character code, which is placed in display storage at the cursor position, causes a blinking marker to appear on the display screen to mark the end of a highlighted field.

4.5. SPECIAL HOST PROCESSOR COMMANDS AND UTS 400 RESPONSES

Various special functions of the UTS 400 can be commanded from the host processor by insertion of special control sequences in text messages. These special functions, their ASCII control codes, and the UTS 400 responses to the control codes are discussed in the following paragraphs.

4.5.1. Send Cursor Address (ESC T)

This command allows the host processor to request the location of the cursor from the UTS 400. Only one ESC T code can be used in a given processor text message.

After receiving the ESC T code from the host processor, the UTS 400 responds to the next traffic poll with a text message containing only the following sequence:

ESC VT R C NUL SI

R and **C** indicate the row and column, respectively, at which the cursor is located. The UTS 400 keyboard is locked and remains locked until the host processor acknowledges receipt of this message.

4.5.2. Call Error Log (ESC P)

This command is used by the host processor to request the error log of the UTS 400 terminal system (master/slave and controller/slave clusters). The text message containing the ESC P code must addressed to the UTS 400 master station or the primary RID SID address.

After receiving the ESC P code from the host processor, the UTS 400 responds to the next traffic poll with the following sequence:

STX ESC VT SP SP NUL SI error log ETX BCC

The error log has priority over any other text messages being sent from the UTS 400.

The error log consists of 23 fields each containing two characters. The following list defines each error log field and gives the order in which the fields are sent. The error count is truncated at 99 for each field.

Error Log Field	Contents
1	8K RAM "3" parity error count
2	8K RAM "2" parity error count
3	8K RAM "1" parity error count
4	8-bit peripheral parity error count
5	7-bit peripheral parity error count
6	Communications parity error count
7	ROM/switch parity error count

Error Log Field	Contents	
8	CPU parity errors	
9	7	
10	6	
11	5	
12	4	Parity errors per display control (display control number)
13	3	
14	2	
15	1	
16	Communications message received with bad character parity	
17	Communications message with wrong BCC received	
18	Communications reply requests issued	
19	Expansion (communications interface) log	
20	7-bit peripheral interface retries required	
21	Expansion (7-bit peripheral) log	
22	8-bit peripheral interface retries required	
23	Expansion (8-bit peripheral) log	

4.5.3. Clear Error Log (ESC R)

This command is used by the host processor to clear the error log of the UTS 400 terminal system (master/slave and controller/slave clusters).

After receiving the ESC R code from the host processor, the UTS 400 resets the error log to zero.

4.5.4. Initiate Confidence Test (ESC Q)

This command is used by the host processor to initiate the confidence test of the UTS 400 terminal system (master/slave and controller/slave clusters).

After receiving the ESC Q code from the host processor, the UTS 400 acknowledges the receipt of the code, waits for an acknowledgment from the host processor, and then initiates the confidence test. The UTS 400 terminal system does not respond to polls while the confidence test is in process.

Optionally, the UTS 400 may respond with a DLE 6 status response to the first poll recognized after the confidence test is complete. (See 3.4.2.4.4.)

4.5.5. Transmit All (ESC DC1)

This command is used by the host processor to request transmission from the UTS 400 of all fields, both protected and unprotected, and associated FCCs. The ESC DC1 code must be sent as the last sequence before the ETX character in a text message from the host processor.

Data sent from the UTS 400 in response to the ESC DC1 code is bracketed between the SOE character (or the home position) and the cursor.

4.5.6. Transmit Variable (DC1)

This command is used by the host processor to request transmission from the UTS 400 of all unprotected fields and associated FCCs. The DC1 code must be sent as the last character before the ETX character in a text message from the host processor.

Data sent from the UTS 400 in response to the DC1 code is bracketed between the SOE character (or the home position) and the cursor.

4.5.7. Transmit Changed (ESC t)

This command is used by the host processor to request transmission from the UTS 400 of only those fields which have been changed and their associated FCCs. The ESC t code must be sent as the last sequence before the ETX character in a text message from the host processor.

Data sent from the UTS 400 in response to the ESC t code is bracketed between the SOE character (or the home position) and the cursor.

4.5.8. Clear Changed (ESC u)

This command is used by the host processor to clear the changed-field designators from selected FCCs in display storage without altering the information in the fields. (Refer to Table 4–2 for detailed information on changed-field designators.)

The ESC u code does not generate a response from the UTS 400.

4.5.9. Automatic Disconnection (DLE EOT)

This command allows the host processor to send the automatic disconnection code to the UTS 400 in an output text message.

The UTS 400, after receiving the DLE EOT code, temporarily drops the data-terminal-ready signal, which causes the modem to disconnect from the communications line. (Refer to 3.10 for a discussion of the interaction between UTS 400 and host processor in an automatic disconnection situation.)

4.5.10. Message Waiting (BEL)

The BEL code is used to send a message-waiting command to the UTS 400. This command causes the UTS 400 MESSAGE WAITING indicator to light and an audible signal to sound. (Refer to 3.4.1 for message format.)

When the UTS 400 operator presses the MSG WAIT key, a request-processor-message sequence is generated by the UTS 400 for transmission to the processor (on a suitable poll); this sequence contains the BEL character, which indicates to the host processor that the MSG WAIT key has been pressed.

4.5.11. Lock Keyboard (DC4 or ESC DC4)

This command makes all keyboard keys inoperative except for the program attention keys. The keyboard remains locked until a text message is received from the host processor or until the UTS 400 operator presses the KBOARD UNLOCK key.

4.5.12. Control Page Access (ESC o)

A specific sequence is used in a text message from the host processor to access the control page and activate the automatic transmit function for the share-type peripheral storage devices. When the automatic transmit function is set in the control page, data retrieved from the share-type peripheral device is transmitted automatically.

The format for setting the automatic transmit function in the 2-line (**PRINT*) field of the control page is given in Figure 4–7.

$$\text{SOH RID SID DID STX } \underbrace{\text{ESC o NUL HT HT HT HT HT HT AT ESC o NUL}}_{\text{Automatic transmit setting sequence}} \text{ ETX BCC}$$

44636

Figure 4–7. Sequence for Control Page Access and Setting of Automatic Transmit Functions

The SID in the message must be for the station (or screen bypass feature)* whose control page is being accessed. Characters in the automatic transmit setting sequence are defined as follows:

ESC o

is the code that accesses the control page. The first occurrence of this code calls the control page to the screen with the cursor located in the first unprotected position of the PRNT field as follows: PRNT(▨). (The complete control page display is shown in 2.4.) The second occurrence of this code returns the control page to storage and sets the control page codes.

NUL

is used for time fill to allow functions to be completed.

**The control page for the screen bypass feature does not appear on any screen but is called from storage and changed by this sequence the same as the control page of any of the display stations.*

HT

is entered six times to tabulate the cursor through the unprotected fields of the control page to the second line of the 2-line (**PRINT*) field, where the automatic transmit code (AT) is inserted as follows:

```
(**PRINT*)
( / /AT)
```

AT

is the code that activates the automatic transmit capability for the share-type peripheral storage devices.

4.6. PERIPHERAL CONTROL

4.6.1. Initiating Peripheral Operation

The UTS 400 operator and the host processor can send data from the UTS 400 screen to a share-type peripheral device by using one of the peripheral initiation commands (Table 4–4). The data to be sent starts with the first character to the right of the SOE character, or with the character in the home position if no SOE character is present, and includes the character at the cursor position. (In peripheral data transfers, the SOE character is always suppressed.) The peripheral initiation command code must be sent as the last sequence before the ETX character in a text message from the host processor.

The data characters actually sent to the share-type peripheral will be determined by the field definition and the initiation command being used.

The UTS 400 operator and the host processor can also solicit data from a share-type peripheral device by using one of these same peripheral initiation commands. Data from the peripheral will be placed on the UTS 400 screen starting at the home position if no unique starting place is defined. A unique starting place can be defined by placing the cursor over an SOE character located at the desired starting position. This procedure suppresses the cursor-to-home and erase-to-end-of-display sequence that is normally sent from the peripheral interface. Data is then placed on the screen starting at the current cursor position. If the data contains FCC sequences or carriage return codes, these codes will reposition the cursor. The type of peripheral initiation command used initially to write the data determines whether the data contains FCCs or carriage returns. The peripheral initiation commands do not alter the data coming from the peripheral device.

NOTE:

The print initiation commands should be used to send data to a printer. The transfer commands will alter the appearance as the data is printed.

Table 4–4. Peripheral Initiation Commands

Command Name and Code	Function	Characteristics of Data Transferred to Peripheral
Print (DC2)	Sends to peripheral all characters from SOE (or home position) to cursor. Also solicits data from a peripheral appropriately selected.	FCC sequences are not included in data stream. Spaces at end of lines are suppressed (except in line containing cursor).
Print Form (ESC H)	Sends to peripheral all unprotected characters from SOE (or home position) to cursor. Also solicits data from a peripheral appropriately selected.	Protected characters are replaced by spaces in data stream. FCC sequences are not included in data stream. Spaces at end of lines are suppressed (except in line containing cursor).
Print Transparent (ESC DC2)	Sends to peripheral all characters from SOE (or home position) to cursor. Also solicits data from a peripheral appropriately selected.	Spaces are not suppressed. FCC sequences are not included in data stream. Cursor returns are suppressed in data stream.
Transfer All (ESC G)	Sends to peripheral all characters from SOE (or home position) to cursor. Also solicits data from a peripheral appropriately selected.	FCC sequences are included in data stream. Spaces at end of fields are suppressed. Spaces at end of lines are suppressed (except in line containing cursor).
Transfer Variable (ESC F)	Sends to peripheral only the variable (not protected) fields between SOE (or home position) and cursor. Also solicits data from a peripheral appropriately selected.	FCC sequences are included for each field transferred. Spaces at end of lines are suppressed (except in line containing cursor). ◀—
Transfer Changed (ESC E)	Sends to peripheral only the changed fields between SOE (or home position) and cursor. Also solicits data from a peripheral appropriately selected.	FCC sequences are included for each changed field transferred. Spaces at end of fields are suppressed. Spaces at end of lines are suppressed (except in line containing cursor).

4.6.2. Control Codes

4.6.2.1. Form Feed (FF)

This control code, when sent to a printer, causes a form feed operation to be performed by the printer. (Refer to Appendix D.) The UTS 400 is transparent (passive) to the form feed function. The FF code is placed in display storage at the cursor position and appears as a blank on the screen.

4.6.2.2. Line Feed (LF)

This control code, when sent to a printer, causes a line feed operation to be performed by the printer. (Refer to Appendix D.) The UTS 400 is transparent (passive) to the line feed function. The LF code is placed in display storage at the cursor position and appears as a blank on the screen.

4.6.2.3. Vertical Tabulator (VT)

This control code, when sent to a printer, causes a vertical tab operation to be performed by the printer. (Refer to Appendix D.) The UTS 400 is transparent (passive) to the vertical tab function if the VT character is not preceded by an escape character. The VT character should not be immediately preceded by an escape character unless a cursor-positioning sequence is intended. The VT code is placed in display storage at the cursor position and appears as a blank on the screen.

4.6.2.4. Carriage Return (CR)

This control code, when sent to a printer, causes a carriage return operation to be performed by the printer. (Refer to Appendix D.) This character is the same as the cursor return character discussed in 4.2.2.3.

4.6.3. Peripheral Programming Considerations

Unique programming considerations for the individual peripherals are presented in the appendixes, beginning with Appendix C.

5. User Programmability

5.1. GENERAL

Programmability is a major feature of the UTS 400 that enables the tailoring of a system to accommodate special user requirements. The standard UTS 400 functions, which are performed by firmware, can be extended, enhanced, and/or modified by incorporating user-prepared programs. Typical functions that may be incorporated through user programming include security checks, data validation, data formatting, arithmetic operations, editing, and text compression.

This section describes the functional characteristics of the UTS 400 programmability feature. Supplemental information describing the UTS 400 instruction set is provided in Appendix F. The program-language manuals referenced in Section 1 should be used in conjunction with the information in this section.

5.2. FIRMWARE SUMMARY

The UTS 400 is an interrupt-driven, serially reentrant, programmed and programmable device with the standard functions performed by firmware stored in read-only memory (ROM). As such, the UTS 400 provides operational capabilities without user programs. The firmware does not require loading before the system will function, nor is it destroyed in the event of a power failure. Further, the firmware does not compete for the random-access memory (RAM) that is provided for storage and execution of user programs.

User programmability complements the firmware by building on its functional base. The UTS 400 firmware—

■ provides the communications protocol,

■ provides screen management,

■ provides for transactions between the UTS 400 and its nonshare- and share-type peripherals, and ←

■ performs operations corresponding to keyboard inputs.

The firmware-controlled protocol is the mechanism for communication between user programs and host processors. Similarly, the standard handlers in the firmware are the means by which user programs communicate with the locally attached UTS 400 nonshare- and share-type peripherals. User programming, therefore, becomes simpler and easier, since user programs need not be concerned with communications protocol or peripheral handling. ←

A summary of the programs and routines in the UTS 400 firmware follows.

The *interrupt handler* handles interrupts and queues activities.

The *dispatcher* checks queues and dispatches available entries to the appropriate handlers.

The *keyboard alphanumeric handler* handles alphanumeric inputs from all keyboards and nonshare-type peripherals. Links to user programs are included to permit user decision on key usage and to accept stored data read from the magnetic stripe of credit-card-type media.

The *keyboard function keys handler* handles function-key inputs from all keyboards. Links to user programs are included to permit user decision on key usage.

The *communications and peripheral input handler* handles communications line or share-type peripheral input data. User code can be incorporated to establish a buffer and then link to this handler to manipulate the data.

The *communications and peripheral output handler* handles communications line or share-type peripheral output data. User code can be incorporated to establish a buffer and then link to this handler using conventional key codes such as transmit, print, and transfer.

User programs operate in conjunction with the standard firmware. The programs are loaded in the UTS 400 RAM, which may include up to 24K bytes of storage. Assembly or compilation of user programs is performed by support software in a host processor. The user programs are loaded in a UTS 400 by means of a downline load mechanism from a host processor and may be either loaded directly into UTS 400 memory or directed to a UTS 400 share-type peripheral for a subsequent offline load.

The UTS 400 firmware provides the following user-program interfaces:

- keyboard/display and nonshare-type peripheral
- communications, and
- share-type peripherals

with each interface consisting of—

- entry points to a user program,
- exit points from a user program to the firmware, and
- flags and pointers.

User programs are written to interface and operate with the support offered by the UTS 400 firmware. The firmware handles all interrupts and dispatches all activities in the UTS 400. Logical links to user programs are provided in the firmware to handle all keyboard input, communications buffers, and peripheral interfaces. Returns are made from user programs to the firmware upon completion of a user-defined task to maintain basic control of the UTS 400.

5.3. OPERATIONAL SUMMARY

After power is applied to the UTS 400, the firmware performs its operations of handling keyboard entries, communications protocol, and peripheral data as triggered by interrupts from the hardware interfaces These interrupts are requests for functions to be performed. Each request is accompanied by data in the form of binary values which identify the specific function. Most of the requested functions must be performed after the interrupt has been handled and interrupts have been reenabled. To enable interrupt handling and to maintain continuous execution of existing operations, the UTS 400 places the data associated with the interrupt and a link to the appropriate routine in a queue. The data and link remain in the queue until the UTS 400 can begin the routine. Searching of queues and dispatching of tasks is performed by the dispatcher in the UTS 400 firmware.

In an idle system, the dispatcher is in a loop monitoring the queues. When the dispatcher finds an entry it jumps to the routine specified by the stored link with the associated data. The rest of the processor registers except the stack pointer, are usable by the routines.

The routines linked in this fashion are the keyboard handlers, the communications handlers, and the peripheral handlers. In these handlers, the passed data is analyzed to determine the function to be performed.

At the appropriate locations in these handlers and in the dispatcher, tests are made to determine if a user program is loaded. If so, the user program will be entered and executed and the user-defined function will be performed, using the data in register B.

There are several entry points into user programs from the firmware. The entry points are at fixed locations at the beginning of the user memory, which starts at octal location 0120000, or hexadecimal (hex) location A000. The user programs must allow for each entry from the firmware even if a direct jump back to the firmware is to be made. There are also fixed exit points in the firmware to which user programs must exit. In addition, there are fixed memory locations which contain information concerning the UTS 400. Some of this information can be modified by user programs; however, this information must be maintained properly if the system is to maintain order. (The system information locations are discussed in detail in 5.5.)

Certain system information locations are referenced by both interrupt and noninterrupt code. Therefore, any changes to this type of data must be performed with interrupts disabled; that is, interrupts must be locked out during the change sequence to ensure the validity of the change. Since disabling interrupts for a long period of time can result in interrupts being lost, it is recommended that the lockout state be continued no longer than 10 instructions.

The setting of a flag maintained by the firmware governs loading of user programs. This flag is cleared when power is applied or when the first block of a program load is received. It is set when the last block of a successful program load has been received, thus enabling execution of the program. The flag can also be cleared by the user program to terminate the program.

5.4. USER PROGRAM MEMORY

NOTE:

The peripheral buffers mentioned in the following discussion pertain to buffers used by the share-type peripherals.

The UTS 400 can contain up to 24K bytes of user RAM, implemented in 8K-byte modules. This RAM may be used for storage of user programs and/or as peripheral buffers.

User RAM is partitioned into 4K-byte segments, thereby providing up to six segments. Each segment may be selected either for storage of user programs or for use as buffers for one or more peripherals. Any combination of segment selections may be made; however, sharing of a segment by both functions is not permissible.

Addressing of the six segments of user RAM is as follows:

Segment	Address (octal)	Address (hex)
1	0120000–0127777	A000–AFFF
2	0130000–0137777	B000–BFFF
3	0140000–0147777	C000–CFFF
4	0150000–0157777	D000–DFFF
5	0160000–0167777	E000–EFFF
6	0170000–0177777	F000–FFFF

Initial selections of segment functions (as user program storage or peripheral buffers) are made by hardware straps. When the UTS 400 is initialized, the segments are defined according to the hardware straps. Although the straps are physical, the implementation of the strapping is logical; that is, during initialization, the UTS 400 firmware examines the strapping and stores the selections in memory.

After initialization, the segment selections may be changed by keyboard entries in the control page. The change is made by specifying reselection in the MM field of the control page in the primary slave or master station. However, reselection can be performed only when no active user programs or buffered peripheral operations are in process.

Reselection entries are defined as follows:

PI	Reassign all user memory according to the initial strapping selections
P7 or P0	Assign all user memory to the peripheral buffer pool
P1	Assign the first segment to user program storage and the remaining segments to peripheral buffers
P2	Assign the first two segments to user program storage and the remaining segments to peripheral buffers
P3	Assign the first three segments to user program storage and the remaining segments to peripheral buffers
P4	Assign the first four segments to user program storage and the remaining segments to peripheral buffers
P5	Assign the first five segments to user program storage and the remaining segments to peripheral buffers
P6	Assign all six segments to user program storage

Pressing the CONTROL PAGE key with reselection entries in the MM field activates the repartitioning sequence. If the user RAM to be reselected is in use, the MM field label in the master-station control page will not change, indicating no reselection. If the reselection entry is acceptable, the first portion of the MM field label will be replaced by the numeric portion of the partition entry.

5.5. INFORMATION LOCATIONS

Certain locations in UTS 400 memory are reserved for storage of information used by the firmware. This information may also be accessed and used by user programs. However, the information must be used only as described in the following paragraphs, because indiscriminate changes may have adverse effects on the operation of the UTS 400.

The following paragraphs contain functional descriptions of the various types of information in the UTS 400 memory. A summary list of the information locations, including the names, addresses, and lengths, is provided in Table 5–1.

Table 5–1. Information Location Summary (Part 1 of 3)

Mnemonic	Name	Address [1] (octal)	Address (Hex)	Length (bytes)	Par. Ref.
ADRFLD	Address field (peripheral device)[5]	[2]+0164	74	6	5.5.2.4.1
BELFLG	Attention key save area	034472	393A	1	5.5.2.3.1
BGCURS	Beginning of screen address	077051	7E29	[4]2	5.5.2.2.1
BGNXTL	Beginning next line	034433	391B	[4]2	5.5.2.2.2
COMDID	Last select DID from host processor	034506	3946	1	5.5.2.3.2
COMIN	User communications input buffer address	035106	3A46	[4]2	5.5.2.3.3
COMINL	Length of message in COMIN	035110	3A48	[4]2	5.5.2.3.4
COMNOC	User communications control byte	035105	3A45	1	5.5.2.3.5
COMOT	User communications output buffer address	035112	3A4A	[4]2	5.5.2.3.6
COMOTL	Length of message in COMOT	035114	3A4C	[4]2	5.5.2.3.7
CRTCTL[3]	CRT control byte	034423	3913	1	5.5.2.2.3
CRTRSV	(Refer to USRCTL)	035074	3A3C	1	5.5.2.2.5
CRTUSR	User-changed screen flag	035073	3A3B	1	5.5.2.2.4
CURSOR	Cursor address	034421	3911	[4]2	5.5.2.2.6
DCFLAG	Control page location flag	034545	3965	1	5.5.2.2.7
DCPAGE	Control page storage area	034547	3967	192	5.5.2.2.8
DSPNDX[3]	Dispatcher index (display index)	076340	7CE0	1	5.5.2.5.1
ERRLOG	Error log locations	076241	7C51	46	5.5.2.5.2
ERSFLG	Screen erase flag	035102	3A42	1	5.5.2.4.2
FDBDCT	Field control character (FCC) generate flag	034436	391E	1	5.5.2.2.9
FDSTAG	FCC staging byte	034437	391F	2	5.5.2.2.10
HOMATT	Dummy FCC for home	034432	391A	1	5.5.2.2.11
LINES	Number of lines on screen	077135	7E5D	1	5.5.2.2.12
LOADFL	Program-loaded flag	077010	7E08	1	5.5.2.5.3
LSTATT	Last attribute	034430	3918	[4]2	5.5.2.2.13
MMTFLD	MM field	[2]+0106	46	2	5.5.2.5.4
MXCURS	End of screen address	077133	7E5B	[4]2	5.5.2.2.14
NUMCHR	Number of characters per line	077137	7E5F	1	5.5.2.2.15

See notes at end of table.

Table 5–1. Information Location Summary (Part 2 of 3)

Mnemonic	Name	Address [1] (octal)	Address (hex)	Length (bytes)	Par. Ref.
OPT2	Screen size	057775	5FFD	1	5.5.1.1
OWNDID	DID to be sent to host processor	034505	3945	1	5.5.2.3.8
OWNSID	SID of station	034445	3925	1	5.5.2.3.9
PERIN	User peripheral input buffer address[5]	035117	3A4F	[4]2	5.5.2.4.3
PERINL	Length of message in PERIN	035121	3A51	[4]2	5.5.2.4.4
PERNOC	User peripheral control byte[5]	035116	3A4E	1	5.5.2.4.5
PEROT	User peripheral output buffer address[5]	035123	3A53	[4]2	5.5.2.4.6
PEROTL	Length of message in PEROT	035125	3A55	[4]2	5.5.2.4.7
POCDSP	POC display locations	076146	7C66	15	5.5.2.5.5
POWRST	Auxiliary present flag	076601	7D81	1	5.5.2.4.8
PRGEND	Program memory end address	077037	7E1F	[4]2	5.5.2.5.6
PRNTFR	Print FROM field	[2]+0142	62	2	5.5.2.4.9
PRNTFU	Print FUNCTION field	[2]+0154	6C	2	5.5.2.4.10
PRNTTO	Print TO field	[2]+0147	67	2	5.5.2.4.11
PRNTYP	Print TYPE field	[2]+045	25	4	5.5.2.4.12
RIDFLD	RID of system	[2]+0253	AB	1	5.5.2.3.10
SEARCH	SEARCH field	[2]+0222	92	23	5.5.2.4.13
SISWTC	External switches	057777	5FFF	1	5.5.1.3
SONALT	Audible alarm (Sonalert) repeat flag	035053	3A2B	1	5.5.2.2.16
STACK	UTS 400 stack area	076144	7C64	100	5.5.2.5.7
STAFLD	STATUS field	[2]+017	F	6	5.5.2.4.14
STRAPS	Strapped DIDs	057760	5FF0	12	5.5.1.2
SYSNDX	System index	076337	7CDF	1	5.5.2.5.8
TIMCNT	Timer interrupt count	077041	7E21	[4]2	5.5.2.5.9
UDISKC	User disk control flag	035127	3A57	1	5.5.2.4.15
USERLC	User-defined locations	035063	3A33	8	5.5.2.5.10

See notes at end of table.

Table 5–1. Information Location Summary (Part 3 of 3)

Mnemonic	Name	Address[1] (octal)	Address (hex)	Length (bytes)	Par. Ref.
USRCTL	Privileged function key indicator	035074	3A3C	1	5.11.3
XFERFR	Transfer FROM field	2+0174	7C	2	5.5.2.4.16
XFERFU	Transfer FUNCTION field	2+0206	86	2	5.5.2.4.17
XFERTO	Transfer TO field	2+0201	81	2	5.5.2.4.18
XFERYP	Transfer TYPE field	2+061	31	4	5.5.2.4.19
XMITYP	Transmit TYPE field	2+075	3D	4	5.5.2.3.11

NOTES:

1. Addresses are subject to change for future firmware releases. Refer to the documentation for the latest release.
2. This address plus either the contents of DCPAGE or the contents of BGCURS (based on DCFLAG).
3. Interrupt and noninterrupt code referenced data.
4. These locations are referenced by using the SHLD and LHLD instructions.
5. These items apply to share-type peripheral usage.

5.5.1. Read-Only Information

The set of read-only memory (ROM) locations contains fixed parameters concerning the UTS 400 configuration. These locations can be read and tested, but a write to a ROM location has no effect on the contents. The various information parameters contained in ROM are presented in 5.5.1.1, 5.5.1.2, and 5.5.1.3.

5.5.1.1. Display Parameter (OPT2)

The display parameter, identified as OPT2, is a flag that defines the screen size. Bits 7 and 6 of the flag identify the screen width and number of lines as follows:

Bits 7–6	Screen Size
00	24 lines, 80 characters/line
01	24 lines, 64 characters/line
10	12 lines, 80 characters/line
11	16 lines, 64 characters/line

5.5.1.2. Peripheral Parameter (STRAPS)

The share-type peripheral parameter is the strapped device identifier (STRAPS), which identifies the DIDs of the system (up to 12) and the associated device types. The STRAPS flag bits are defined as follows:

Bit 7	= 1, 8-bit auxiliary device = 0, 7-bit auxiliary device
Bit 6	= 1, read and write device = 0, write-only device
Bits 7–6	= 00, mnemonic P for communications output printer (COP) or terminal printer = 01, mnemonic C for tape cassette system = 10, mnemonic T for 8-bit printer = 11, mnemonic D for diskette subsystem
Bit 5	= 1, retryable device = 0, nonretryable device
Bit 4	= 1, select drive 2 = 0, select drive 1
Bits 3–0	= the lower 4 bits of the DID, identifying the function. If bit 6 is a binary 1, this is the write DID; the read DID is 1 greater.

5.5.1.3. General Parameters

The general parameters are defined by bits derived from the external switches flag (SISWTC). The flag bits are defined as follows:

Bit 7	= 1, monitoring enabled = 0, monitoring disabled
Bit 6	= 1, protect mode = 0, FCC mode

5.5.2. Read and Write Information

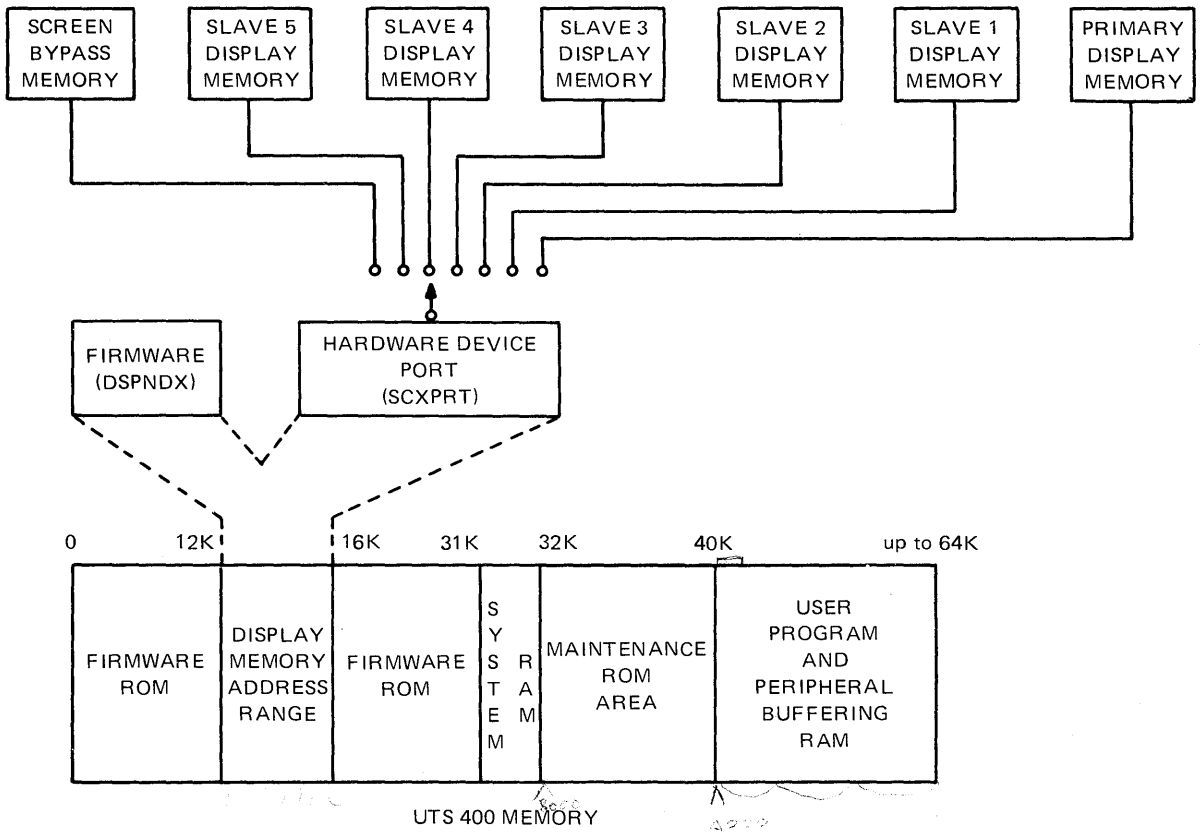
The random-access memory (RAM) locations contain information such as pointers, flags, and tables. The contents of these locations provide information used by the firmware; therefore, indiscriminate changes to the contents should be avoided, because such changes may cause the UTS 400 to malfunction and produce conditions that can be corrected only by reinitialization.

There are three RAM areas in the UTS 400, as follows:

- User program RAM, which is defined in 5.4.
- Display-independent memory, starting at octal location 076000 (hex 7C00) and ending at 077777 (7FFF). This area, which is part of the standard UTS 400 memory, is used to store information concerning the system and is independent of any display.

- Display-dependent memory, starting at octal location 030000 (hex 3000) and ending at 037777 (3FFF). This area, which is also part of the standard UTS 400 memory, is used to store information for particular displays. There are as many display memories (plus the screen bypass memory) in a UTS 400 system as there are displays. However, only one set of addresses is available. An indexing mechanism is therefore provided to accommodate the multiple memories. Figure 5–1 illustrates this method of indexing, which is performed via the firmware and a hardware device port. The firmware sets the dispatcher index (DSPNDX, described in 5.5.2.5.1) to correspond to the hardware scope index port (SCXPRT, described in 5.6.4) and thus selects the appropriate display memory.

The various parameters contained in RAM are divided into functional groups as presented in 5.5.2.1 through 5.5.2.5.



44645

Figure 5–1. UTS 400 Memory and Display Memory Indexing

5.5.2.1. Display Format Data

The largest section of the display-dependent memory is the refresh portion, which stores the data to be displayed on the screen of the associated station. In addition, the display memory contains bytes of data that are used for controlling the format of the screen and the display data. All of the locations in the display memory contain either display data or control data. Maintenance of the screen is dependent upon the maintenance of this memory.

The display storage starts at octal location 030000 (hex 3000) and ends at 034420 (3910). The part of the memory in which data is stored for display is determined by the screen format, as illustrated in Figures 5–2 through 5–5. These figures illustrate the initialization of the four possible screen formats. (Up to 17 dummy locations are included in this storage area, as shown in the figures.) Display parameters BGCURS, MXCURS, NUMCHR, and OPT2 contain information associated with formatting this memory.

The display memory can be visualized as containing 24 groups of 96 bytes, each group representing one line on a screen. The 96 bytes in a group include the 80 or 64 displayable characters and 16 FCC positions. At least one FCC position will always be present as an end-of-line indicator. For a 64-character screen format, the 64 characters are preceded by 16 nondisplayable positions.

Upon initialization, the last 16 bytes of each group are FCC positions, which are binary 0's. When FCCs are generated, leading FCC positions are deleted as the data to the right of the cursor in the line is shifted right one place to make room for the FCC. The process is reversed when FCCs are removed.

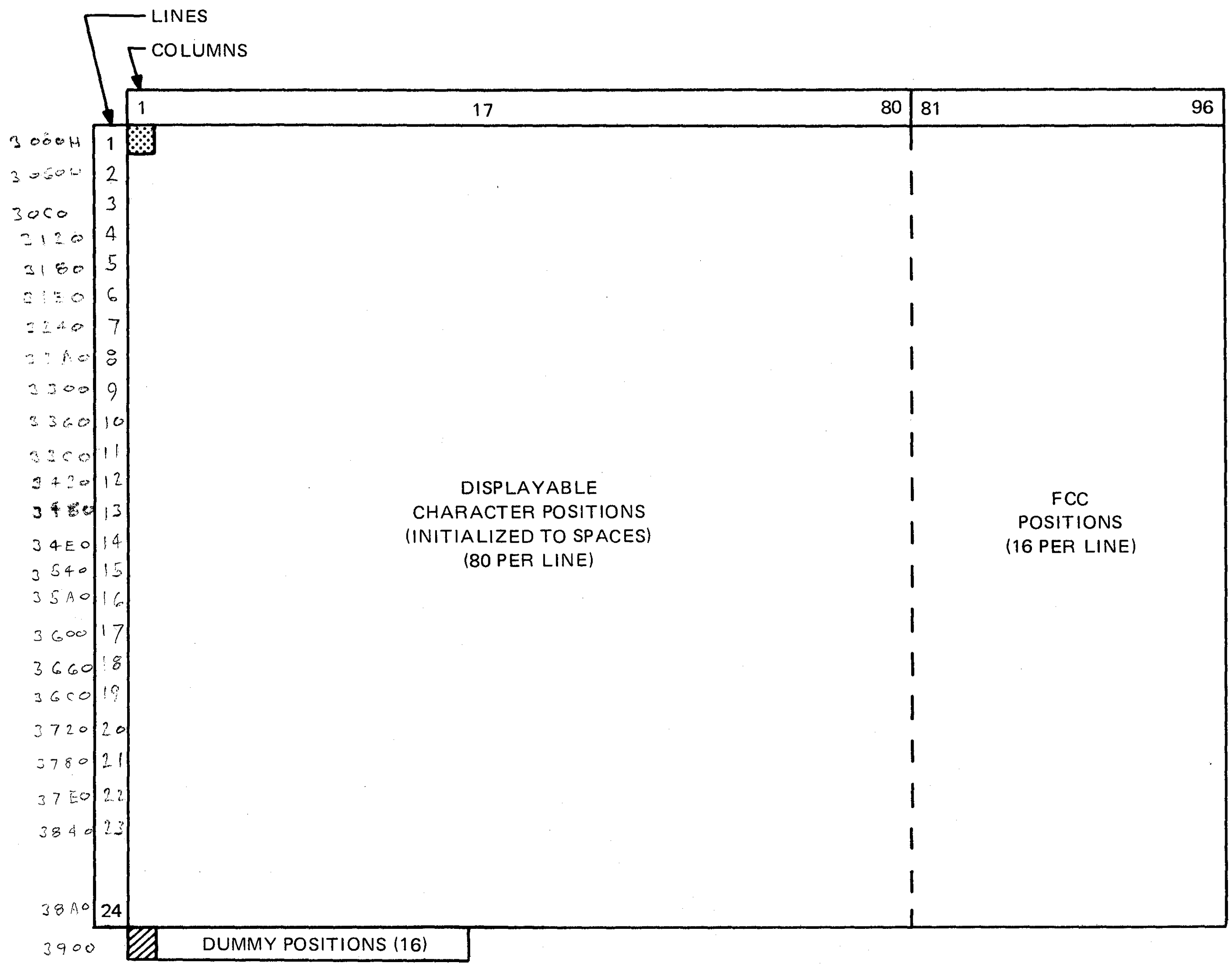
As FCCs are generated and removed, the count of FCCs and FCC positions must remain at 16. The result is thus a line consisting of 80 or 64 displayable characters (and 16 deleted positions for a 64-character screen format), from 0 to 15 FCCs, and from 16 to 1 FCC positions.

Reducing the number of lines is accomplished by excluding certain of the groups. This is performed by adjusting the values in the MXCURS and BGCURS parameters. The excluded groups are initialized to octal location 040 in display memory and are not displayable.

Display memory locations are nine bits wide plus a parity bit. Since the processor registers are only eight bits wide, two steps are required to obtain all nine bits. The ninth bit is not used by the firmware in the nonrefresh portion of the display memory, but all nine bits are used in the refresh portion. Bits 8 and 6 through 0 are data bits, while bit 7 is the FCC flag.

If bit 7 is a binary 0, the byte is a data character and bits 8 and 6 through 0 define the character. If bit 7 is a binary 1, the byte is an FCC and bits 8 and 6 through 0 define the FCC. The remaining eight bits of an FCC byte are defined as follows:

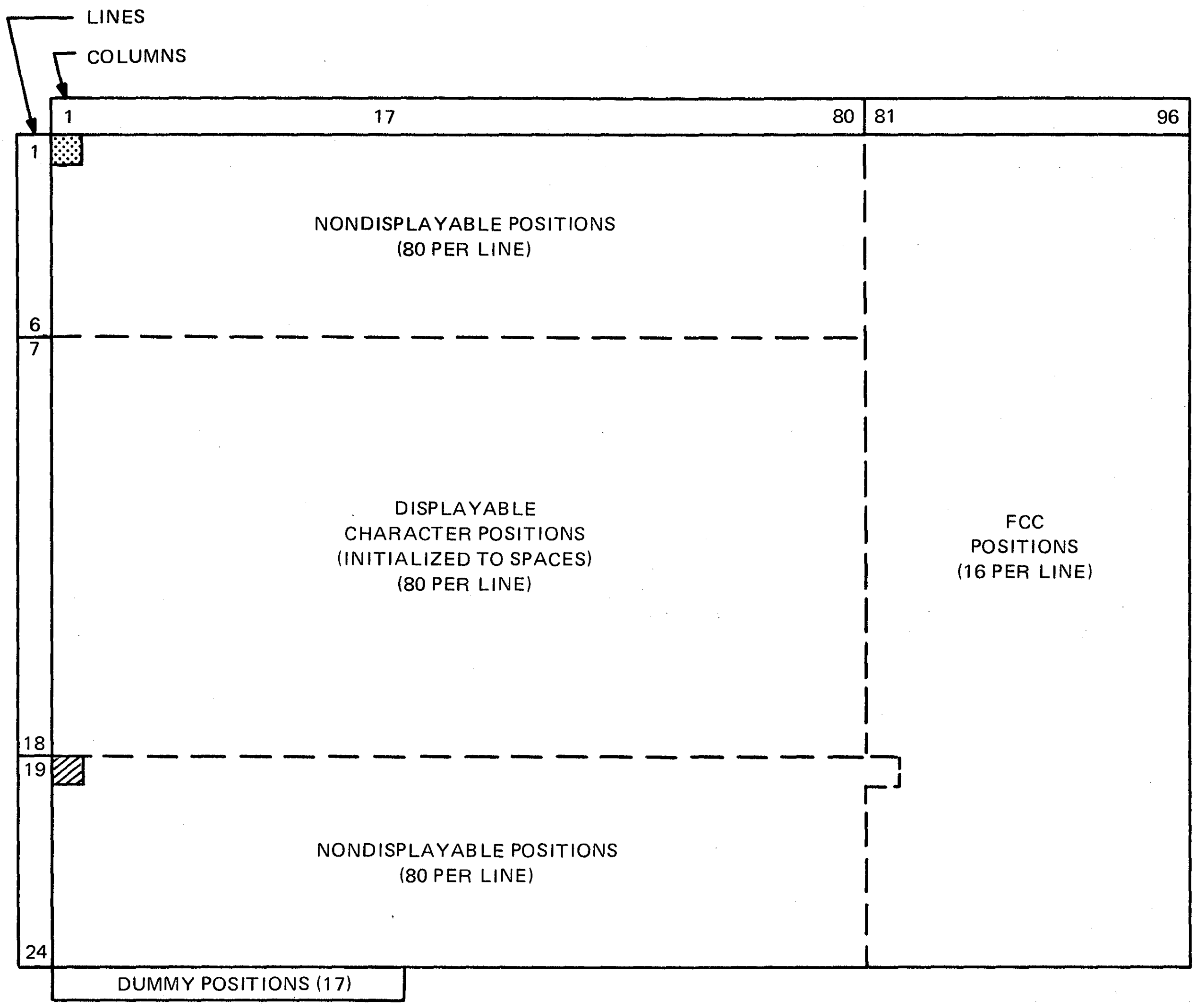
Bit 8	= 1, FCC does not contain tab stop = 0, FCC contains tab stop
Bit 6	= 1, field not changed = 0, field changed
Bits 5–4	= 00, normal display intensity = 01, display intensity off = 10, low display intensity = 11, blinking display
Bit 3	= user-program defined. Set to binary 0 by the firmware when FCC is generated from the keyboard.
Bit 2	= 1, field right-justified = 0, field not justified
Bits 1–0	= 00, any input allowed = 01, alphabetic input only = 10, numeric input only = 11, protected (bit 2 must be binary 0)



⛶ – FIRST DISPLAY MEMORY LOCATION
▨ – DUMMY FCC; OCTAL 0300
DUMMY POSITIONS – OCTAL 000
FCC POSITIONS – OCTAL 000

44646

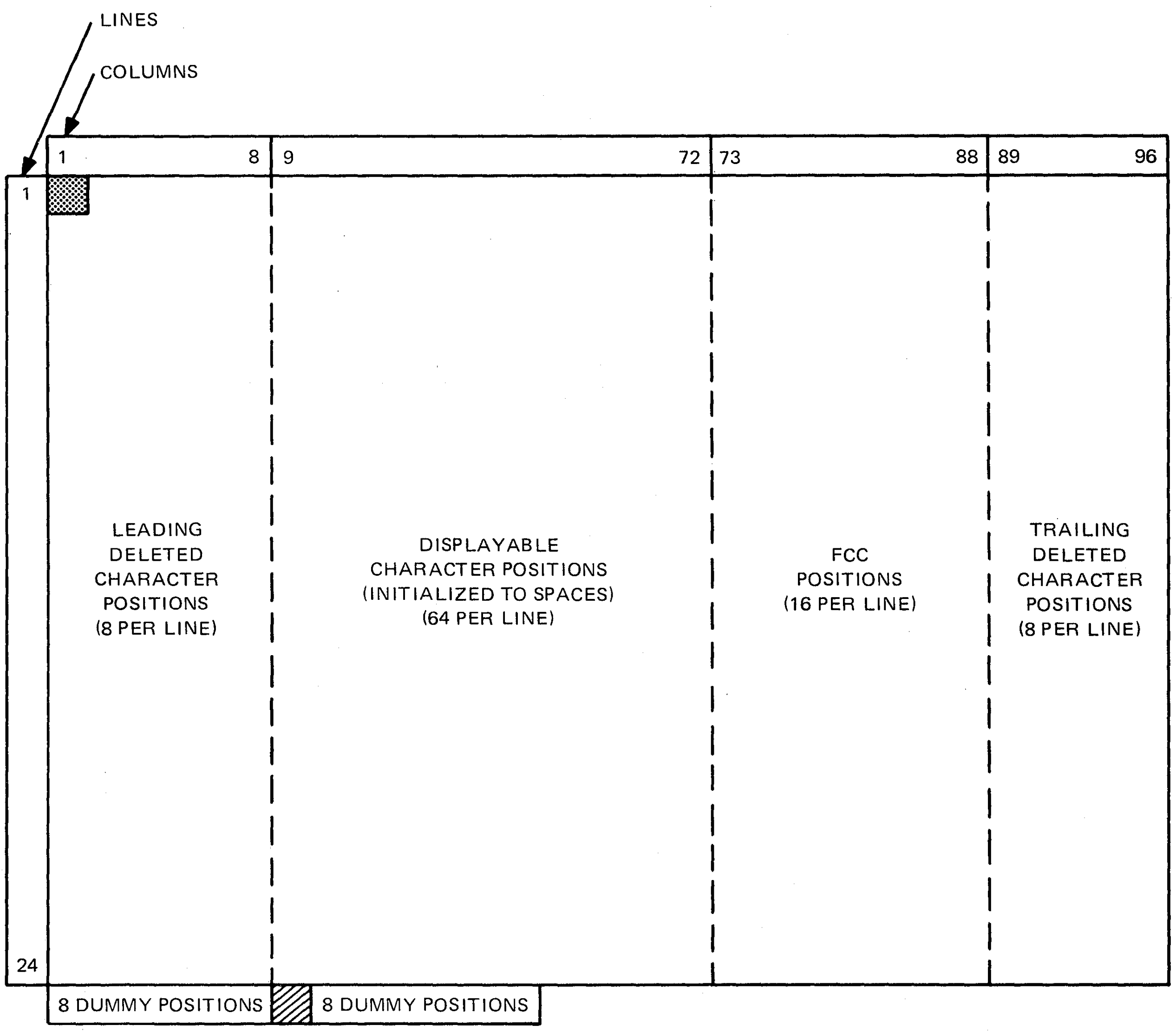
Figure 5–2. Display Memory Initialized for 24 by 80 Format


LINES
COLUMNS
1
17
80
81
96
1
6
7
18
19
24
NONDISPLAYABLE POSITIONS (80 PER LINE)
DISPLAYABLE CHARACTER POSITIONS (INITIALIZED TO SPACES) (80 PER LINE)
FCC POSITIONS (16 PER LINE)
NONDISPLAYABLE POSITIONS (80 PER LINE)
DUMMY POSITIONS (17)

░ – FIRST DISPLAY MEMORY LOCATION
▨ – DUMMY FCC; OCTAL 0300
DUMMY POSITIONS – OCTAL 000
FCC POSITIONS – OCTAL 000
NONDISPLAYABLE POSITIONS – OCTAL 040 (SPACE)

44652

Figure 5–3. Display Memory Initialized for 12 by 80 Format


LINES
COLUMNS
1
8
9
72
73
88
89
96
1
24
LEADING DELETED CHARACTER POSITIONS (8 PER LINE)
DISPLAYABLE CHARACTER POSITIONS (INITIALIZED TO SPACES) (64 PER LINE)
FCC POSITIONS (16 PER LINE)
TRAILING DELETED CHARACTER POSITIONS (8 PER LINE)
8 DUMMY POSITIONS
8 DUMMY POSITIONS

▦ – FIRST DISPLAY MEMORY LOCATION
▨ – DUMMY FCC; OCTAL 0300
DUMMY POSITIONS – OCTAL 000
FCC POSITIONS – OCTAL 000
DELETED CHARACTER POSITIONS – OCTAL 001

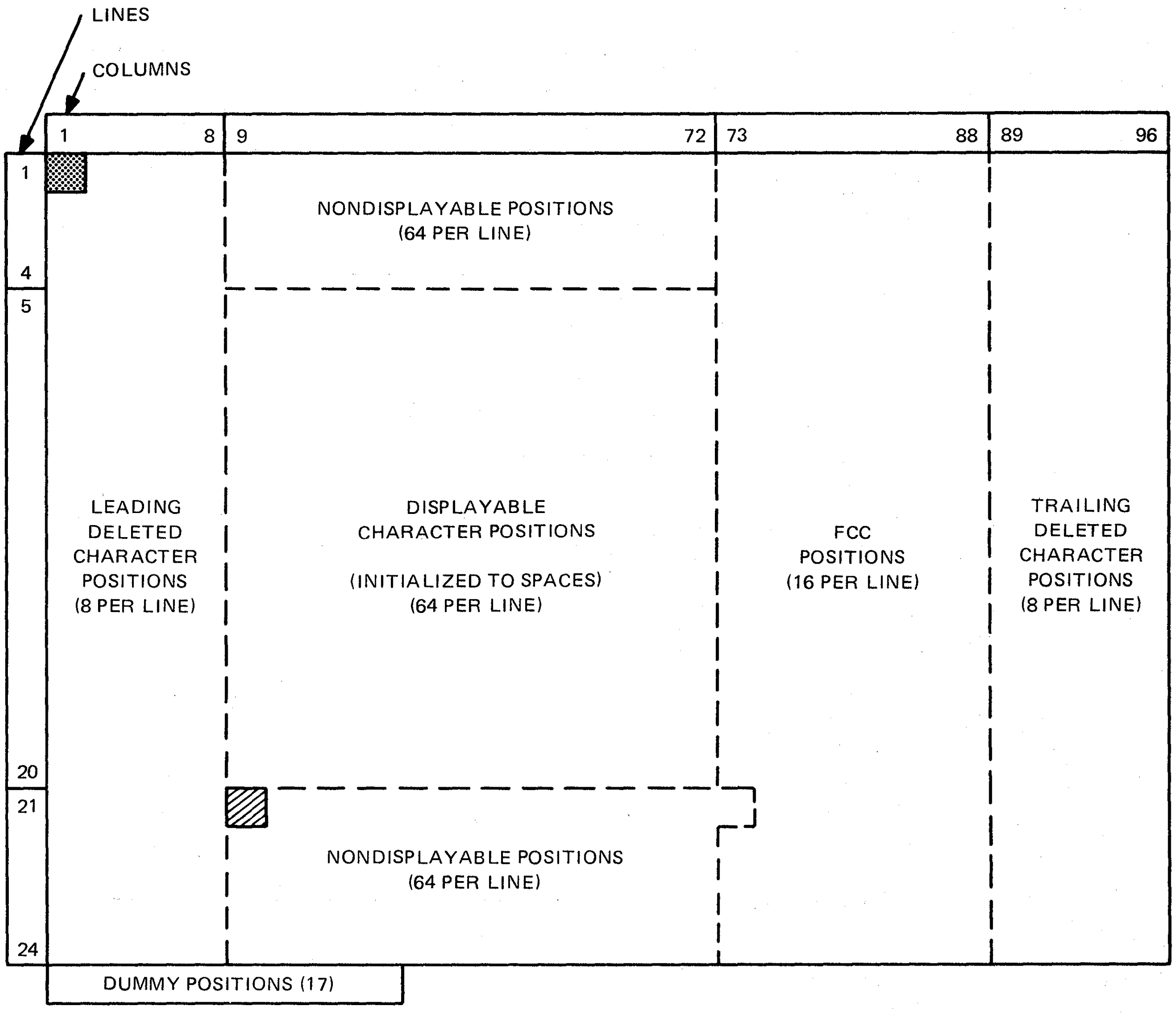
44653

Figure 5–4. Display Memory Initialized for 24 by 64 Format

5.5.2.2. Display Pointers and Flags

The binary 0's, binary 1's, and FCCs in the display memory function as flags to the program as the screen data is manipulated. An additional set of locations contains parameters concerning the display memory. These parameters are used and updated as needed by the firmware.

The display pointers and flags are defined in the following paragraphs.


LINES
COLUMNS
1
8
9
72
73
88
89
96
1
4
5
20
21
24
NONDISPLAYABLE POSITIONS (64 PER LINE)
LEADING DELETED CHARACTER POSITIONS (8 PER LINE)
DISPLAYABLE CHARACTER POSITIONS (INITIALIZED TO SPACES) (64 PER LINE)
FCC POSITIONS (16 PER LINE)
TRAILING DELETED CHARACTER POSITIONS (8 PER LINE)
NONDISPLAYABLE POSITIONS (64 PER LINE)
DUMMY POSITIONS (17)

▩ –FIRST DISPLAY MEMORY LOCATION
▨ – DUMMY FCC; OCTAL 0300
DUMMY POSITIONS – OCTAL 000
FCC POSITIONS – OCTAL 000
NONDISPLAYABLE POSITIONS – OCTAL 040 (SPACES)

44654

Figure 5–5. Display Memory Initialized for 16 by 64 Format

5.5.2.2.1. Beginning of Screen Address (BGCURS)

The BGCURS flag defines the home address of the display, which is determined by the screen size. This flag also defines the first location of an ordered line table consisting of up to 24 addresses (2 bytes per address). The first address in the table is the home address. Successive addresses define the start of subsequent lines of the screen.s,

5.5.2.2.2. Beginning of Next Line (BGNXTL)

The BGNXTL parameter is a pointer to refresh memory that identifies the address of the beginning of the line immediately below the cursor. When the cursor is positioned in the last line of the screen, the BGNXTL value is the same as that of the MXCURS parameter. The BGNXTL value is the address of the first byte of a 96-byte group on an 80-character screen, or the seventeenth byte on a 64-character screen.

This value must be updated any time the address in the CURSOR flag is changed in such a way that the cursor is moved out of the line from which it started.

The BGNXTL pointer should not be set with the line 1 address. A malfunction will occur if this is done.

5.5.2.2.3. CRT Control Byte (CRTCTL)

The CRTCTL flag controls the audible alarm and display indicators of the UTS 400. The condition of the CRTCTL flag bits determines the state of the UTS 400 as follows:

Bit 7	= 1, WAIT indicator turned on (keyboard locked) = 0, WAIT indicator turned off (keyboard unlocked)
Bit 6	= 1, display of cursor disabled = 0, display of cursor enabled
Bit 5	Not used
Bit 4	= 1, POLL indicator turned on momentarily = 0, normal state of bit
Bit 3	= 1, AUXILIARY BUSY indicator turned on = 0, AUXILIARY BUSY indicator turned off
Bit 2	= 1, MESSAGE WAITING indicator turned on = 0, MESSAGE WAITING indicator turned off
Bit 1	= 1, MESSAGE INCOMPLETE indicator turned on = 0, MESSAGE INCOMPLETE indicator turned off
Bit 0	= 1, audible alarm sounds momentarily = 0, normal state of bit

5.5.2.2.4. User-Changed Screen (CRTUSR)

The CRTUSR flag must be set whenever the user program stores data in the display memory or changes the display memory in any way (by entering FCCs, for example) without using firmware facilities. The flag must also be set if the address in the CURSOR flag is changed by the user's program, even if the rest of the display is unchanged. It does not have to be set when the user program changes display pointers that are above display memory location octal 34422 (hex 3912). The flag will be cleared by firmware as necessary.

When the least significant bit of the CRTUSR flag is binary 1 and all other bits are binary 0's, the flag is set.

5.5.2.2.5. Reserved Display Memory (CRTRSV)

This display memory byte is now assigned as USRCTL (5.11.3).

5.5.2.2.6. Cursor Address (CURSOR)

When bit 6 of the CRTCTL flag is a binary 0, a cursor is generated based on the value in the CURSOR flag. The hardware, when refreshing the display, checks that the address in display memory from which it is reading matches the value in the CURSOR flag. If so, access is made to the cursor generation logic and a cursor is displayed.

If the address defines a location that does not contain data to be displayed, a match will not occur and the cursor will not be displayed. Therefore, care should be exercised in placing an address in the CURSOR flag which is not a displayable position of display memory. The address should not point to an FCC, to an FCC position, to the binary 1's for a 64-character screen format, or to an area outside the refresh portion of the display memory.

5.5.2.2.7. Control Page Location Flag (DCFLAG)

This flag identifies the location of the control page. If the DCFLAG content is a binary 0, the control page is in memory. If the content is a binary 1, the control page is on the screen. Any other values are illegal.

5.5.2.2.8. Control Page Storage Area (DCPAGE)

DCPAGE is the address of the control page when the contents of DCFLAG are binary 0's. The control page occupies 192 bytes, starting at this address. If the contents of DCFLAG are binary 1's, the address of the control page is found in the field named BGCURS. Note that BGCURS is a 2-byte field that contains the address of the control page (when DCFLAG contains binary 1's), and DCPAGE is the beginning control page location itself (when DCFLAG contains binary 0's).

All of the fields in the control page are assigned a value relative to either the contents of the DCPAGE flag or the contents of the BGCURS flag, depending on the contents of DCFLAG.

Refer to UP-8358 for a definition of the control page.

5.5.2.2.9. FCC Generate Flag (FDBDCT)

When the FCC GENERATE key function is processed by the firmware, the data on the screen is shifted to make room for the field control character (FCC), and the last character on the line is temporarily shifted off and saved. When the FCC is placed in memory, the last character is restored to the screen. An illegal character in the FCC generation sequence will cause the original screen display to be restored, if required.

The FDBDCT flag is set to a count of 5 when the FCC generation operation starts. Each legal key depression decrements the contents of the flag as the operator proceeds through the sequence. When the FDBDCT content is a binary 0, the firmware is not in an FCC generation sequence. This flag should be checked by the user program to ensure that key depressions in the FCC generation sequence are separated from all others and the FCC generation sequence is maintained.

5.5.2.2.10. FCC Staging Byte (FDSTAG)

The FDSTAG flag contains the accumulation of the bits making up the FCC as the FCC generation sequence progresses. The FDSTAG+1 location, containing the tab bit, is initially set to an octal value of 010 (hex 8) when the sequence starts. On the first FCC sequence entry (intensity), the FDSTAG flag is set to the appropriate value for the desired intensity, the change-indicator bit is set for no change, and the FCC bit is reset to an octal value of 0360 (hex F0), 0340 (E0), 0320 (D0), or 0300 (C0). The FDBDCT contents are decremented with each succeeding step.

On the next entry, the tab bit in the FDSTAG+1 location is either unchanged or set to a binary 1. With the next entry (field type), an OR operation is performed on bits 1 and 0 and the contents of FDSTAG as required by the input. The next entry (right justify) causes the same operation on bit 2. The last entry causes the contents of FDSTAG+1 to be output to the scope status port (5.6.2) and causes the FDSTAG contents to be written to memory at the address specified by the CURSOR flag. The cursor is moved to the first character of the field and the last character on the line is restored.

The FDSTAG contents remain nonzero and serve as the flag for disabling FCCs. The FCC enable function sets this flag to a binary 0. User programs can use this byte as a flag for FCCs being disabled.

The user program can use the FDBDCT and FDSTAG flags to expand the sequence and include one more entry by setting bit 3. The FDBDCT flag should not be changed by user programs.

5.5.2.2.11. Dummy FCC for Home (HOMATT)

For the firmware routines that use the change-indicator bit in the FCC, it is important to have an FCC defined for every field. To ensure this, a dummy FCC, octal 0300 (hex C0), is stored at the HOMATT location. This dummy FCC is used as the default FCC for the first field on the screen when there is no user-generated FCC at the home position. Use of this location, since it is not physically at the beginning of the screen, requires extra checks during certain operations.

5.5.2.2.12. Number of Lines on Screen (LINES)

The LINES flag defines the number of lines on the screen (12, 16, or 24).

5.5.2.2.13. Last Attribute (LSTATT)

The LSTATT pointer contains the address of the first FCC to the left of the cursor position. This value is not maintained during a scan operation and is reset when the cursor scan key is released by the return from the SCNDFM exit (see 5.7.3.2). The field where the cursor is located is determined by the FCC to which the address in LSTATT points.

5.5.2.2.14. End of Screen Address (MXCURS)

The MXCURS parameter is a flag that defines the first address past the end of the screen. A dummy FCC, octal 0300 (hex C0), is maintained at the location identified by this flag.

5.5.2.2.15. Number of Characters per Line (NUMCHR)

The NUMCHR parameter is a flag that defines the width of the screen in numbers of displayable characters (80 or 64).

5.5.2.2.16. Audible Alarm (Sonalert) Repeat Flag (SONALT)

The SONALT flag controls the repeated sounding of the audible alarm. This byte is set to a binary 1 to enable repeated sounding of the alarm and, conversely, to a binary 0 to disable the repeated sounding of the alarm. A keyboard unlock command clears the SONALT flag.

5.5.2.3. Communications Pointers and Flags

These locations provide information necessary for host processor communication. The information is used and updated by the firmware as data is sent to and received from the host processor.

The communications pointers and flags are defined in the following paragraphs.

5.5.2.3.1. Attention Key Save Area (BELFLG)

The BELFLG location contains the attention key code or message code that is to be sent to the host processor. A code is placed in the BELFLG location by the firmware when one of the attention keys (F1 through F22) or the MSG WAIT key has been pressed by the UTS 400 operator and a user program is not active. The user program may place a code for one of attention keys F5 through F22 in the BELFLG location by loading the key code in register B and exiting to the ATTNFM exit (see 5.7.3.2).

If the value in the BELFLG location is a binary 0, there are no previous codes to be sent. The user program reads this location to determine if there is a code to be sent. If there is a code, any entry to ATTNFM will result in the new code being discarded. To ensure that a particular code is sent to the host processor, a check of the BELFLG location can be made to determine when the value changes to a binary 0. At that time, a jump can be made to the ATTNFM exit with the code to be sent to the host processor.

5.5.2.3.2. Last Select DID From Host Processor (COMDID)

The COMDID location contains the last received device identifier (DID) which was greater than octal 0161 (hex 71). This is the DID that will be used by the firmware to select a peripheral device as a host-processor-initiated operation is started. The contents of this location should not be changed by a user program.

5.5.2.3.3. User Communications Input Buffer Address (COMIN)

The COMIN location contains a user-defined address of a buffer in user memory where text input for a station will be stored instead of being displayed on a screen. The data stored will be all the characters (except nulls) between the STX and ETX. None of the characters stored in this buffer are interpreted by the firmware. Thus, if a print command is in the message, the command will be placed in the buffer without starting a share-type peripheral operation. An acknowledgment will be sent for the message instead of the wait-before-transmit (WABT) response.

When the firmware gives the user program control at the TEXTCM entry point (as discussed in 5.7.2.2), the keyboard is locked. The keyboard may be unlocked, after the text message in the COMIN-defined buffer has been processed, by the appropriate setting of the CRTCTL flag. (See 5.5.2.2.3.)

5.5.2.3.4. Length of Message in COMIN (COMINL)

At the beginning of an operation associated with the COMIN buffer, the value in the COMINL location is the maximum length of the user-defined COMIN buffer. As the message is received, the count of the characters in the message is stored in this location.

If the message is longer than the buffer, the overflow characters will be lost.

5.5.2.3.5. User Communications Control Byte (COMNOC)

The COMNOC flag indicates that a user program is to perform input and/or output using user memory as a buffer. If bit 0 is a binary 1 when a transmit command is issued, the contents of the COMOT and COMOTL locations (see 5.5.2.3.6 and 5.5.2.3.7) are used in performing the output. When bit 7 is a binary 1, the contents of the COMIN and COMINL locations are used for input of the next message received. Once an operation involving one of these buffers has been completed, the corresponding bit is cleared. The user program must reset the bit to enable subsequent use of the COMNOC flag.

5.5.2.3.6. User Communications Output Buffer Address (COMOT)

The COMOT location contains a user-defined address of a buffer in user memory where characters will be obtained for output to a station. The data in this buffer comprises the message between the STX and ETX.

5.5.2.3.7. Length of Message in COMOT (COMOTL)

The value in the COMOTL location contains the user-defined length of the message contained in the COMOT buffer.

5.5.2.3.8. DID To Be Sent to Host Processor (OWNDID)

The OWNDID location contains the DID that will be sent to the host processor in messages from a station. The OWNDID value is initially set at octal 0160 (hex 70) and is changed to that of the received DID, which is greater than octal 0160, or to octal 0177 (hex 7F) when an offline share-type peripheral operation is started. The most recent of these operations determines the value in this location.

5.5.2.3.9. SID of Station (OWNSID)

The OWNSID location contains the station identifier (SID) of a station. The user program should not change the contents of this location.

5.5.2.3.10. RID of System (RIDFLD)

The RIDFLD location contains the remote identifier (RID) of the system. The user program should not change the contents of this location.

5.5.2.3.11. Transmit TYPE Field (XMITYP)

The XMITYP flag defines the beginning of the transmit-operation TYPE field within the control page.

5.5.2.4. Peripheral Pointers and Flags

The information in the share-type peripheral pointers and flags is used and updated by the firmware as necessary to perform read, write, and other share-type peripheral operations. These pointers and flags are defined in the following paragraphs.

5.5.2.4.1. Address Field (ADRFLD)

The ADRFLD flag defines the beginning of the address field within the control page. The report-address and copy operations update the information in the address field. The contents of this field can be read by the user program to obtain an address following either of these operations. Since the field is used to show the results of an operation, and since there is no carryover of information between operations, the user program can also use the address field as a temporary display field.

5.5.2.4.2. Screen Erase Flag (ERSFLG)

This flag directs the share-type peripheral handler to execute or discard the cursor-to-home and erase-display sequence in the input data stream from the peripheral. If this flag is set to a binary 0, the cursor-to-home and erase-display sequence will be executed if the peripheral returns the sequence to the firmware. If the flag is not a binary 0, the sequence will be discarded. The peripheral has the same discard capability during read operations based on the cursor being over the SOE. In this case, the flag is used to allow the discard capability on a read associated with a search operation.

On completion of any share-type peripheral operation, the ERSFLG flag is set to a binary 0. The user program must reset the flag if it is to be active again.

This flag is ignored when the input is to user memory. However, if the cursor is over an SOE during a read operation to user memory, the cursor-to-home and erase-display sequence is discarded. There is no capability for a read with search and discard of this sequence when reading to user memory.

5.5.2.4.3. User Peripheral Input Buffer Address (PERIN)

The PERIN pointer contains a user-defined address that points to a buffer in user memory where share-type peripheral text is stored when so indicated by the PERNOC flag (5.5.2.4.5). The text placed in the user buffer is not interpreted by the firmware. All characters with a numeric value of an ETX and less will have been removed from the text.

The text from the 7-bit interface will be ASCII characters, and bit 7 will be a binary 0. Katakana characters from this interface will be bracketed by SO and SI as they are on the communications interface.

Characters from the 8-bit interface will be stored exactly as read from the interface. Katakana characters from this interface will be indicated by bit 7 being set to a binary 1.

5.5.2.4.4. Length of Message in PERIN (PERINL)

At the beginning of an operation associated with the PERIN buffer, the value in the PERINL flag is the maximum length of the user-defined PERIN buffer. When a message is received, the count of the characters in the message is stored in the PERINL location.

If the message is longer than the buffer, the overflow characters will be lost.

5.5.2.4.5. User Peripheral Control Byte (PERNOC)

The PERNOC flag indicates that a user program is to perform share-type peripheral input and/or output, using user memory as a buffer. If bit 0 is a binary 1 when a print or transfer command is issued, the contents of the PEROT and PEROTL locations (see 5.5.2.4.6 and 5.5.2.4.7) are used in performing the output. When bit 7 is a binary 1, the contents of the PERIN and PERINL locations are used for input. Once an operation involving one of these buffers has been completed, the corresponding bit is cleared. The user program must reset the bit to enable subsequent use of the PERNOC flag.

5.5.2.4.6. User Peripheral Output Buffer Address (PEROT)

The PEROT pointer contains a user-defined address of a buffer in user memory where characters will be obtained for a write operation to a station. The lower 7 bits are used for the 7-bit interface, and all eight bits are used for the 8-bit interface.

NOTE:

When the share-type peripheral device is a diskette subsystem, a binary zero (null) sent to the device will be stored on the diskette; but when that block is read, the null will cause the read sequence to be terminated if the read destination is the screen. If the destination is a user buffer, the null will cause the rest of that sector to be skipped but the read will resume on the subsequent sectors until the count in PERINL is satisfied or until the block end-sentinel is encountered. If a null is to be stored on a diskette, use the UDISKC flag (5.5.2.4.15) and perform one sector write and read.

When a single-sector read is used, the null and all subsequent data in the sector are read into the user buffer.

5.5.2.4.7. Length of Message in PEROT (PEROTL)

The value in the PEROTL flag is the user-defined length of the message in the PEROT buffer.

5.5.2.4.8. Auxiliary Present Flag (POWRST)

The POWRST flag identifies the presence or absence of a working peripheral interface. With bit 0 set to a binary 1, an 8-bit interface is present. With bit 7 set to a binary 1, a 7-bit interface is present. All the other bits are binary 0's.

5.5.2.4.9. Print FROM Field (PRNTFR)

The PRNTFR flag defines the beginning of the print-operation FROM field within the control page.

5.5.2.4.10. Print FUNCTION Field (PRNTFU)

The PRNTFU flag defines the beginning of the print-operation FUNCTION field within the control page.

5.5.2.4.11. Print TO Field (PRNTTO)

The PRNTTO flag defines the beginning of the print-operation TO field within the control page.

5.5.2.4.12. Print TYPE Field (PRNTYP)

The PRNTYP flag defines the beginning of the print-operation TYPE field within the control page.

5.5.2.4.13. SEARCH Field (SEARCH)

The SEARCH flag defines the beginning of the SEARCH field within the control page.

5.5.2.4.14. STATUS Field (STAFLD)

The STAFLD flag defines the beginning of the STATUS field within the control page.

5.5.2.4.15. User Disk Control Flag (UDISKC)

The UDISKC flag indicates that a user program is to perform input and/output to a disk without firmware formatting of the disk sectors. Bits 0 and 5 of the flag control a write operation, while bit 7 controls a read. The flag can be examined only when corresponding operation bits in the PERNOC flag are set. The value in the PERIN or PEROT pointer defines the buffer address. The buffer length, however, is set to 128 bytes; since the operation is a sector read or write, the length specified in the PERINL or PEROTL flag is not used.

If the UDISKC contents are a binary 0, even though the PERNOC flag may be set to use user memory, read and write operations will be formatted with the first byte of each sector being used as a sector sequence indicator within a block. Only up to 127 bytes of each sector will be used for data.

A user program may specify that a write to disk be a control sector write. This is indicated by setting bits 5 and 0 as follows:

Bits 5 and 0	Operation
01	Normal write
11	Control sector write

A control sector write is used when compatibility with other vendor equipment is required.

5.5.2.4.16. Transfer FROM Field (XFERFR)

The XFERFR flag defines the beginning of the transfer-operation FROM field within the control page.

5.5.2.4.17. Transfer FUNCTION Field (XFERFU)

The XFERFU flag defines the beginning of the transfer-operation FUNCTION field within the control page.

5.5.2.4.18. Transfer TO Field (XFERTO)

The XFERTO flag defines the beginning of the transfer-operation TO field within the control page.

5.5.2.4.19. Transfer TYPE Field (XFERYP)

The XFERYP flag defines the beginning of the transfer-operation TYPE field within the control page.

5.5.2.5. General Pointers and Flags

This set of locations stores information concerning the state of the system. The information is used and updated as the firmware performs its various functions. The pointers and flags are defined in the following paragraphs.

5.5.2.5.1. Dispatcher Index (DSPNDX)

The DSPNDX location contains a value identifying a particular display memory to be used in an operation. The firmware interrupt code uses this value to select a display memory as part of the restore-environment code before returning to the interrupted code.

As an entry is made to the user program from the firmware, the DSPNDX and SCXPRT contents are set to reflect the display memory from which the operation was initiated. (The USIDLE entry point, described in 5.7.2.4, is an exception.)

When a user program changes the DSPNDX value to access a display memory, the SCXPRT value (see 5.6.4) must be changed to match.

5.5.2.5.2. Error Log Locations (ERRLOG)

The ERRLOG locations contain the error log maintained by the system. The ERRLOG content is a set of decimal counts that can be obtained upon request from the host processor or operator. The error log is presented as a continuous string of ASCII characters (octal values 060 through 071 or hex 30 through 39), starting at the ERRLOG location, with the sets arranged in the following order:

ERRLOG	Third 8K RAM internal parity error
+2	Second 8K RAM internal parity error
+4	First 8K RAM internal parity error
+6	8-bit peripheral internal parity error
+8	7-bit peripheral internal parity error
+10	Communications internal parity error
+12	8K ROM internal parity error
+14	CPU internal parity error

+16	Screen bypass memory
+18	Display 6 internal parity error
+20	Display 5 internal parity error
+22	Display 4 internal parity error
+24	Display 3 internal parity error
+26	Display 2 internal parity error
+28	Display 1 internal parity error
+30	Communications input character parity error
+32	Communications input BCC error
+34	Communications reply request sent count
+36	Communications extra
+38	7-bit peripheral retry count
+40	7-bit peripheral extra
+42	8-bit peripheral error count
+44	8-bit peripheral extra

5.5.2.5.3. Program-Loaded Flag (LOADFL)

The LOADFL flag informs the firmware that a user program is or is not loaded. If the flag is a binary 0, the firmware does not enter the user program area. If the flag is not a binary 0, entries are made to the user program. This flag is initialized to a binary 0 and is set by the receipt of the transfer address during a successful load sequence. The LOADFL flag is set to a binary 0 by:

1. the receipt of an ESC SO sequence in a host processor load message (see 5.9.1),
2. pressing of the PROGRAM LOAD key, or
3. the user program, prior to an exit to the dispatcher if the user program is to be terminated.

The user program can inhibit the loading of another user program by storing an octal value of 0252 (hex AA) in the LOADFL flag.

5.5.2.5.4. MM Field (MMTFLD)

The MMTFLD flag defines the beginning of the MM field within the control page. The flag is examined by the firmware and the indicated functions are performed as the control page is moved from the screen to display memory.

5.5.2.5.5. POC Display Locations (POCDSP)

The POCDSP locations contain the test characters which are displayed on the screen at the end of the power-on confidence (POC) test. The displayed characters identify the modules in the system and the module status. A slash (/) means the module corresponding to that location is not in the system; an asterisk (*) means the module is in the system and operable; and a hexadecimal number means the module is in the system but has been found in error. The value of the number represents the detected error. Starting at the POCDSP location, the sets are arranged in the following order:

POCDSP	Third 8K RAM
+1	Second 8K RAM
+2	First 8K RAM
+3	8-bit peripheral interface
+4	7-bit peripheral interface
+5	Communications
+6	8K ROM
+7	CPU
+8	Screen bypass memory
+9	Display 6
+10	Display 5
+11	Display 4
+12	Display 3
+13	Display 2
+14	Display 1

5.5.2.5.6. Program Memory End Address (PRGEND)

The PRGEND flag contains the address of the end of the user program memory (see 5.4). This address should not be changed by the user program.

5.5.2.5.7. UTS 400 Stack Area (STACK)

The STACK flag contains the address of the last byte of the stack. The user should exercise caution not to exceed the limits of the stack or unknown results will occur. Further, the user should not use any of the stack pointer instructions. Only the CALL, RET, PUSH, and POP instructions are to be used in conjunction with the stack.

5.5.2.5.8. System Index (SYSNDX)

The SYSNDX flag contains a value identifying the last display memory configured in the system. Binary 1's are included for configured display memories. A binary 0 identifies the first nonexistent display memory when the byte is scanned from right to left. For example, an octal value of 0357 (hex EF) indicates that four display memories are configured.

The value in the SYSNDX flag is used by the firmware as the start value in display memory scan loops.

5.5.2.5.9. Timer Interrupt Count (TIMCNT)

The TIMCNT location contains a 16-bit count that is incremented each time the firmware receives a timer interrupt. The timer interrupts occur approximately every 100 milliseconds. User programs should not write to this location unless interrupts are disabled.

5.5.2.5.10. User-Defined Locations (USERLC)

The USERLC flag defines a set of locations that has been set aside for use by user programs. The contents of these locations are not defined by the firmware. These locations are special in that they are in the display memory and are provided to assist the user in producing reentrant code.

5.6. DEVICE PORTS

Device ports are registers that provide the links between the firmware and the hardware. These ports are accessed through the use of the IN and OUT instructions in the processor. The device ports that may be used by user programs are described in 5.6.1 through 5.6.4. A summary list of the device ports and their addresses is provided in Table 5–2.

Table 5–2. Summary of Device Port Addresses

Mnemonic	Device Port Name	Address (octal)	Address (hex)
RABPRT	RAM boundary port	073	3B
RASPRT	RAM status port	070	38
SCSPRT	Scope status port	000	0
SCXPRT	Scope index port	001	1

5.6.1. RAM Boundary Port

The user RAM locations are nine bits wide; therefore, special handling is required for the extra bit. The extra bit is accommodated by use of the RAM boundary port (RABPRT). Since the user RAM stores data and program instructions, it is necessary to identify to the hardware where in memory the extra bit is to be used. Failure to do so will result in the extra bit being modified as instructions are read by the processor for execution.

The RAM boundary port contains a value which defines the data boundary of a user program in user RAM. The contents are the upper eight bits of a user-defined address in user RAM. The upper eight bits of an operand address are compared with the RABPRT contents on a memory read or write. If the address is equal to or greater than the RABPRT contents, the ninth bit will be transferred to or from the RAM status port (5.6.3).

The RAM boundary port also provides write protection for the user program. Any location less than the address in this port is treated the same as ROM; that is, a write to memory locations less than the address in the RAM boundary port will not affect data in those locations. Locations equal to or greater than the address in the RAM boundary port may be read and written as in standard RAM.

5.6.2. Scope Status Port

The scope status port (SCSPRT) stores the extra bit of the last location read from a display memory. A write to the display memory uses the extra-bit value to set the extra bit in memory.

In the scope status port, the extra bit is bit 7. When the contents of this port are read (using an IN instruction), the remaining seven bits are undefined; when written (using an OUT instruction), the remaining seven bits must be set to an octal 010 (hex 8) pattern to keep the maintenance lamp on.

5.6.3. RAM Status Ports

The RAM status port (RASPRT) stores the extra bit of the last location read from user RAM whose address is equal to or greater than the value in the RAM boundary port (5.6.1). A write to an address uses the extra bit value in this port to set the extra bit in memory. This bit is set by a write to this port using an OUT instruction.

In the RAM status port, the extra bit is bit 7. When the contents of this port are read (using an IN instruction), the remaining seven bits are undefined; when written (using an OUT instruction), the remaining seven bits must be set to an all-binary-0's pattern.

5.6.4. Scope Index Port

The scope index port (SCXPRT) operates in conjunction with the dispatcher index (5.5.2.5) as an index for the display memories in a UTS 400 system. When a memory location is addressed that is within the display memory address range, all display memories examine the SCXPRT contents to determine which is selected.

If a display memory is not selected, no action is taken by the memory. If a display memory is selected, the memory responds by performing a read or write as indicated.

In the UTS 400, only one display memory should be selected at a time. A copy of a byte written to the scope index port must be stored in the dispatcher index (DSPNDX) to identify the station selected.

5.7. USER PROGRAM INTERFACES

The user program interfaces consist of entry points from the firmware to user programs and exit points from user programs to the firmware. Entry points are the means by which the UTS 400 firmware passes control to user programs. These points are at fixed locations in user RAM, separated by at least three addresses so that jump instructions can be included at each entry point.

Exit points are jumps from user programs back to the UTS 400 firmware and are also at fixed locations. Several options are available when returning control to the firmware, as follows:

- Exit to the firmware dispatcher with no further processing.
- Exit to the firmware with data in register B for further processing.
- Perform required processing and exit to the firmware at another firmware routine.

Descriptions of the initialization process, the entry points, and the exit points are provided in 5.7.1, 5.7.2, and 5.7.3, respectively. A summary list of the entry and exit points and corresponding addresses is provided in Table 5–3.

NOTE:

In the character-protection mode when the user program exits to the firmware to process a function, that function is processed according to the functional description provided in Appendix G.

Table 5–3. Summary of Entry and Exit Points

Name	Entry Points			Exit Points			Remarks
	Mnemonic	Address		Mnemonic	Address		
		Octal	Hex		Octal	Hex	
Attention Key	ATTNKB	0120000	A000	ATTNFM	000024	14	User program must lock/unlock keyboard as required; see 5.7.2.
Function Key	FUNCKB	0120003	A003	FUNCFM	000016	0E	
Normal Key	NORMKB	0120006	A006	NORMFM	000005	05	
Scan Key	SCANKB	0120011	A009	SCANFM	000021	11	
Scan Key Released	SCNDKB	0120014	A00C	SCNDFM	000013	0B	
Communications Handler	TEXTCM	0120017	A00F	–	–		See 5.7.3.3.
Peripheral Handler	TEXTPR	0120022	A012	–	–		See 5.7.3.4.
System Idle	USIDLE	0120025	A015	–	–		
Dispatcher	–	–	–	DISP	000054	2C	
Find FCC to Left of Cursor	–	–	–	FNDATT	000037	1F	See 5.7.3.5.
Error Count Increment	–	–	–	INCSUB	040066	4036	See 5.7.3.5.

5.7.1. Initialization

Upon completion of a successful user program load, the LOADFL flag is set (5.5.2.5.3) and the user program is entered at the address specified in the last program load block. This address, defined as the transfer address, is the start address of the user program to be executed. Using this entry mechanism permits initialization of the user-program pointers, tables, and flags as required.

Control is passed with the selected master or primary slave station whose SID was in the program load block. During initialization, the station keyboard is in a locked-out state. The scheduled user program should unlock the station keyboard.

5.7.2. Entry Points

5.7.2.1. Entry From the Keyboard Handler

There are five entry points to a user program from the UTS 400 keyboard handler, as follows:

- Normal key entry (NORMKB)
- Function key entry (FUNCKB)
- Cursor scan key entry (SCANKB)
- Cursor-scan-key-released entry (SCNDKB)
- Attention key entry (ATTNKB)

Entry at certain of these points locks the keyboard, causing the WAIT indicator to light and all subsequent key depressions (except keyboard unlock) to be discarded until the keyboard is unlocked. The user program is responsible for unlocking the keyboard when handling of the key entry has been completed.

Entries that do not lock the keyboard should be handled in a period of time not exceeding 10 milliseconds. If this time is exceeded, the queues may be filled with subsequent key depressions.

The following six key inputs are not passed to user programs:

MESSAGE WAITING
KBOARD UNLOCK
Attention keys F1 through F4

These six keys are required to maintain control and provide override capability for the UTS 400. The remaining attention keys (F5 through F22) can be used as entries.

Entries from the keyboard handler are detailed as follows:

- Normal key entry (NORMKB) – The keys and corresponding codes passed in register B are listed in Table 5–4.
- Attention key entry (ATTNKB) – The keys and corresponding codes passed in register B are listed in Table 5–5.

- Function key entry (FUNCKB) – The keys and corresponding codes passed in register B are listed in Table 5–6.
- Cursor scan key entry (SCANKB) – The keys and corresponding codes passed in register B are listed in Table 5–7.
- Cursor-scan-key-released entry (SCNDKB) – This entry is a notification that cursor scanning has been terminated. A return must be made to the SCNDFM exit (5.7.3.2). The data passed in register B is not meaningful when this entry is used.

Table 5–4. Normal Keys and Codes

Key(s)	ASCII Code	Octal Code	Hex Code
Alphanumeric and special	–	040–0177	20–7F
Katakana	–	0240–0377	A0–FF
Carriage return	CR	015	D
Tab forward*	HT	011	9
Line feed	LF	012	A
Form feed	FF	014	C
Start-of-entry (SOE)	RS	036	1E

*Keyboard locked

Table 5–5. Attention Keys and Codes

Attention Key	ASCII Code	Octal Code	Hex Code
F5*	SP	040	20
F6*	!	041	21
F7*	"	042	22
F8*	#	043	23
F9*	$	044	24
F10*	%	045	25
F11*	&	046	26
F12*	'	047	27
F13*	(	050	28
F14*	)	051	29
F15*	*	052	2A
F16*	+	053	2B
F17*	,	054	2C
F18*	–	055	2D
F19*	.	056	2E
F20*	/	057	2F
F21*	o	060	30
F22*	1	061	31

*Keyboard locked

Table 5–6. Function Keys and Codes

Function Key	ASCII Code	Octal Code	Hex Code
Erase to end of display*	a	0141	61
Erase to end of line*	b	0142	62
Erase to end of field*	K	0113	4B
Erase display (all)*	M	0115	4D
Delete in line*	c	0143	63
Delete in display*	C	0103	43
Delete line*	k	0153	6B
Insert in line*	d	0144	64
Insert in display*	D	0104	44
Insert line*	j	0152	6A
Backspace*	g	0147	67
Forward space*	h	0150	68
Tab stop set	HT	011	9
Cursor to home	e	0145	65
Backward tab*	z	0172	7A
Line duplicate*	y	0171	79
Clear changed bits*	u	0165	75
Report address*	l	0154	6C
Control page*	o	0157	6F
Transmit*	p	0160	70
Print*	q	0161	71
Transfer*	r	0162	72
Status*	N	0116	4E
Hang up*	O	0117	4F
Load program*	S	0123	53
FCC clear*	w	0167	77
FCC locate*	x	0170	78
FCC reenable*	v	0166	76
FCC generate*	s	0163	73
Search cassette*	n	0156	6E
Backward one block*	m	0155	6D
Release auxiliary buffer*	I	0111	49
Recover auxiliary buffer*	J	0112	50

*Keyboard locked

Table 5–7. Cursor Scan Keys and Codes

Cursor Scan Key*	ASCII Code	Octal Code	Hex Code
Scan left	g	0147	67
Scan right	h	0150	68
Scan down	i	0151	69
Scan up	f	0146	66

*The keyboard is not locked for these entries

5.7.2.2. Entry From the Communications Handler

The UTS 400 firmware enters a user program from the communications handler at a point defined as the user communications text input completion (TEXTCM). This entry provides notice of text input to the user program so that the user program can perform operations triggered by completion of the input.

Entry is directed by a host processor text message that does not contain a delayed function such as a share-type peripheral operation or transmission. (Messages containing delayed functions are handled as if the user program were not loaded. Messages containing a DC4 code as the last character of text are also handled in the same manner.) Data in the host processor message will be placed in the display memory with all the functions received in the text, such as delete line or erase display, having been performed.

In the case of input to user memory, delayed functions are ignored by the UTS 400 firmware. The user program is responsible for processing delayed functions such as print.

When the firmware gives the user program control to process text in user memory, the keyboard is locked. The keyboard may be unlocked, after the text message in user memory has been processed, by the appropriate setting of the CRTCTL flag. (See 5.5.2.2.3.)

When entered from the communications handler, the user program should return to the firmware dispatcher at the DISP exit.

Data passed in register B is not meaningful with the TEXTCM entry.

5.7.2.3. Entry From the Peripheral Handler

After completion of any offline-initiated share-type peripheral function, the UTS 400 firmware enters a user program from the peripheral handler at a point defined as the user peripheral function completion (TEXTPR). Information concerning the operation will be in the appropriate control page location.

When entered from the peripheral handler, the user program should return to the firmware dispatcher at the DISP exit.

Data passed in register B is not meaningful when the TEXTPR entry is used.

5.7.2.4. Entry From System Idle

The UTS 400 firmware dispatcher enters a user program at this point when there are no entries in the system queues. The system idle (USIDLE) entry enables the execution of lengthy user programs that exceed the suggested maximum UTS 400 program execution time of 100 milliseconds.

A display memory will be selected at USIDLE entry. However, the display memory selection at the time of entry may not be the one required for the operation. Therefore, the user program must, as part of the initial operations, make the proper display memory selection.

When a USIDLE entry is used, the user program should return to the firmware dispatcher at the DISP exit.

The data passed in register B is not meaningful with the USIDLE entry.

5.7.3. Exit Points

5.7.3.1. Exit to the Dispatcher

When a user program has completed a processing function and no further processing is to be performed, the user program returns to the firmware dispatcher at the DISP exit. The dispatcher is therefore free to schedule other tasks in the queues or to schedule a system idle entry (5.7.2.4) if no entries are in the queues. The keyboard must be unlocked before an exit is made (see 5.8.2.2.2) unless further processing is required to complete a function.

Data passed in register B is not meaningful when this exit is used.

5.7.3.2. Exit to the Keyboard Handler

There are five exits from a user program to the keyboard handler, one for each entry from the keyboard handler to a user program. If no processing is required at an entry, an exit is made directly to the firmware at the appropriate exit point.

Data passed during entry to a user program initiates the appropriate processing. Should a user program receive control with data that does not lock out the keyboard and if the data to be returned requires a keyboard lockout, the user program must first lock out the keyboard. In the opposite case, the user program must unlock the keyboard (see 5.8.2.2.2).

The exits to the keyboard handler are summarized as follows:

- Normal key exit (NORMFM) – The operation indicated by the data in register B is performed when this exit is used (see Table 5–4).

- Function key exit (FUNCFM) – The operation indicated by the data in register B is performed when this exit is used (see Table 5–6). In addition, the keyboard unlock code (octal 0114, or hex 4C) can be passed to this exit to unlock the keyboard.

- Cursor scan key exit (SCANFM) – An exit should be made at this point only when a user program is entered at the SCANKB entry, described in 5.7.2.1 (see Table 5–7).

- Cursor-scan-key-released exit (SCNDFM) – Pointers not maintained while a cursor scan operation is in progress are reset at this exit. Because a user program may perform other operations that require the same pointers, this exit may be used as desired. The LSTATT pointer (5.5.2.2.13) is reset, and the cursor is moved out of a protected field if it is in one or is moved to the beginning of the field if it is in a right-justified field.

- Attention key exit (ATTNFM) – This exit causes ATTNKB codes (5.7.2.1) to be sent to a host processor in the method defined for program attention keys or to be discarded if there is already an attention key code to be sent. (See Table 5–5 for keys and codes.)

↓

NOTE:

In the character-protection mode, processing for the normal-key, function-key, and cursor-scan-key-released exits is performed according to the functional description provided in Appendix G. Processing for the cursor-scan-key and attention-key exits, under the same terminal conditions, is performed as if the terminal were operating in normal mode.

↑

5.7.3.3. Exit to the Communications Handler

Communications functions are performed according to the current selections made by an operator or by a user program in the control page. Each operation is initiated by a user program exit, at the function key exit (FUNCFM), using the code in register B for a transmission command. The screen is used as the source of transmission functions for each exit to this handler unless the COMNOC flag (5.5.2.3.5) is set for user buffering.

5.7.3.4. Exit to the Peripheral Handler

Share-type peripheral functions are performed according to the current selections made by an operator or by a user program in the control page. Each operation is initiated by a user program exit, at the function key exit (FUNCFM), using the code in register B as a print, transfer, or other command to initiate a peripheral function. The share-type peripheral functions use the screen data unless the PERNOC flag is set for user program buffering.

5.7.3.5. Exit to Firmware Subroutines

Exits are provided from user programs to two firmware subroutines that may be used by user programs. The firmware is accessed by the user program with a CALL instruction and, upon completion of processing in the firmware, a return is made to the calling user program. The two firmware subroutines that may be used in this manner are described in the following paragraphs.

5.7.3.5.1. Find FCC to Left of Cursor (FNDATT)

This subroutine scans left from the cursor address to locate an FCC and sets the LSTATT pointer (5.5.2.2.13) with the FCC address. If an FCC is not located, the address in the HOMATT location is used. When this subroutine is entered, the cursor address must be in registers H and L. Registers A, D, and E are used by the subroutine. When the return is made to the user program, the LSTATT pointer is set and registers H and L contain the address stored in LSTATT.

5.7.3.5.2. Error Count Increment (INCSUB)

This subroutine, which is used by the firmware to increment the error logs of the UTS 400 (see 5.5.2.5.2), can be used by a user program for a similar operation. Operationally, the subroutine receives the address of a 2-digit (2-byte) decimal number and increments the number unless the number is 99, in which case no incrementing is performed. When the subroutine is entered, the address of the second of the two bytes is in registers H and L. Registers A, H, and L are used by the routine but no data is returned.

5.8. USER PROGRAMMING AIDS

Previous paragraphs in this section have covered individual aspects of UTS 400 programmability such as the information locations, device ports, and user program interfaces. This paragraph contains information to assist in efficient use of the UTS 400 programmability feature. The use of these programmability aspects is defined and examples concerning preparation of user programs are provided.

User programs, when implemented, must accommodate all actions initiated by the hardware interfaces. The hardware interfaces are categorized as follows:

- Keyboard (includes nonshare-type peripheral)
- Communications input/output
- Share-type peripheral storage/retrieval

A user program is normally a routine or set of routines designed to perform processing based on the operations in the UTS 400. In line with the hardware interface categories, user programs are categorized into groups as follows:

- Those that process keyboard key depressions
- Those that process read-only, nonshare-type peripheral input
- Those that process messages for communications or peripheral interfaces
- Those that perform processing when the UTS 400 is in an idle mode

In user programs, symbolic references should be made to all information locations, since the absolute addresses are subject to change between firmware releases. Using symbolic references, user programs may be executed regardless of the firmware release by merely performing a reassembly of the user programs.

Two mechanisms in the UTS 400 facilitate efficient checkout of user programs:

- Reinitializing to the original state by use of the TEST switch
- Obtaining a program dump to assist in problem isolation

Sets of entry points from the UTS 400 firmware are provided for each of the user program categories, with corresponding exit points from user programs to the UTS 400 firmware. Information locations and device ports are available to facilitate efficient user program preparation. The use of entry and exit points, information locations, and device ports is presented in 5.8.1, 5.8.2, and 5.8.3, respectively.

5.8.1. Entry and Exit Points

User programs, when implemented, must accommodate all entry points as defined in 5.7. That is, each entry point must contain a jump to a user program or a direct jump to an exit point, and these jumps must be established by the user program. Those few restrictions applying to proper use of exit versus entry points are defined in 5.7.

When a user program is not implemented, the UTS 400 firmware automatically bypasses the entry and exit points.

Control is passed from the firmware to user programs via an entry point table. As the firmware receives interrupts from the hardware interfaces, the firmware is directed to a location in the entry point table. For example, when a keyboard key is pressed, the firmware jumps to the appropriate location in the entry point table; thereafter, processing for this key depends on the code in this table location.

Following is an example entry point table:

Entry	Action	To	Description
ATTNKB	Jump	ATTNPR	Jump to user program identified as ATTNPR
FUNCKB	Jump	FUNCPR	Jump to user program identified as FUNCPR
NORMKB	Jump	NORMPR	Jump to user program identified as NORMPR
SCANKB	Jump	SCANPR	Jump to user program identified as SCANPR
SCNDKB	Jump	SCNDPR	Jump to user program identified as SCNDPR
TEXTCM	Jump	DISP	Return to the firmware dispatcher
TEXTPR	Jump	PERPHL	Jump to user program identified as PERPHL

In this example, all entry points except TEXTCM indicate a jump to a user program. The user program in this example would process all functions that occur offline to the host processor.

As a jump is made to a user program (for example, to ATTNPR), the user program may appear as follows:

Routine	Instruction	Description
ATTNPR:	MOV A, B	Begin user program
	. . .	Intervening instructions
	JMP DISP	Processing completed
	or	
	JMP ATTNFM	Firmware to complete processing

For this example, the user program processes the attention key and returns control to the firmware dispatcher or provides a jump to the ATTNFM exit for completion of the processing, using the code in register B. The code in register B may be the same as or different than that originally passed to the entry.

The suggested maximum period for processing any entry is 100 milliseconds, which permits execution of approximately 20,000 instructions. If this time is not adequate, the user program can lock the keyboard, store the environment, and return to the firmware dispatcher (DISP). When there are no entries in the dispatcher queues, control is passed to the user program at the USIDLE entry point for completion of the processing.

5.8.1.1. Key Processing

The user program is entered for each keyboard key depression. Therefore, it is suggested that user programs process each key code only to the extent that following key codes can be processed in an orderly manner.

5.8.1.2. Message Processing

Message processing is performed when communications-line or share-type peripheral operations have been completed (except in the case of output to the communications line). The user program is entered once for each successful text input message or each peripheral operation.

The entry point for communications-line input text messages is TEXTCM. Entry is made only when a text message has been received correctly. Upon completion of message processing by the user program, a return is made to the firmware dispatcher at the DISP exit.

NOTE:

The user program is not notified when output to the communications line has been completed. Receipt of a subsequent input message indicates completion of the previous output.

An entry to the user program based on completion of share-type peripheral operations (read, write, search, report address, report status, backward one block, or buffer release/recover) is made at the TEXTPR entry. The normal return from the user program, after completion of processing, is to the firmware dispatcher at the DISP exit. To determine the validity of a peripheral operation, the user program must check the control page status.

The initiation of communications output and share-type peripheral operations is performed by loading the desired function key code (such as transmit, transfer, print, or search) into register B and exiting to the firmware at the FUNCFM exit. The control page for the station, as well as the communications or peripheral pointers and flags, should have been set to the desired values prior to the exit.

Following is an example sequence for a communications input at the TEXTCM entry:

Label	Action	Description
	–	
TEXTCM	Jump to TEXTIN	Communications portion of entry point table
	–	
	. . .	TEXTIN instructions
	Jump to DISP	Return control to firmware dispatcher

5.8.1.3. Idle System Entry

The idle system entry (USIDLE) is the last one in the entry point table. Consequently, this entry location may contain a jump to another location in a user program, or it may contain the initial instruction of a user program. Following execution of the user program, an exit to the firmware dispatcher (at DISP) should be made when a USIDLE entry is used.

When control is passed to the user program at this entry point, the dispatcher index and the scope index port must be set to the particular display memory desired. User programs cannot rely on these two pointers being set to a specific value upon entry.

5.8.2. Information Location Usage

The UTS 400, being a programmed terminal, contains information in memory that is used to maintain proper operational control. This information may also be used in preparing user programs for execution in the UTS 400. The stored information is divided into two major classes: read-only information, and read/write information.

The read-only information is stored in ROM and can be read and tested, but a write to a ROM location has no effect on the contents.

Read/write information is stored in RAM and contains items such as pointers, flags, and tables.

Since information locations in RAM can be read and written, indiscriminate or uncontrolled changes to these locations must be avoided. User program changes in RAM locations must be made under carefully controlled conditions to avoid adversely affecting the operational integrity of the UTS 400. Accessing and changing information during program checkout is not of great concern, however, since the system can be reinitialized to restore the original information.

5.8.2.1. Read-Only Information

The read-only information consists of three types of parameters:

- Screen size (OPT2) – Defines the screen size of the UTS 400.

- Device identification (STRAPS) – Contains the identifiers for all share-type peripheral devices attached to a UTS 400 cluster. There are 12 bytes for a cluster, each byte containing information for a device. Using this data, user programs can identify the devices with which they are communicating.

- External switch selections (SISWTC) – Contains information indicating the positions of the MONITOR and FCC PROTECT switches.

User programs may access this read-only information as desired to obtain the indicated information. Detailed descriptions of the information in ROM can be found in 5.5.1.

5.8.2.2. Read and Write Information

The contents and characteristics of the five functional groups of read/write information contained in UTS 400 RAM are described in 5.5.2. The use of this information in user programs is summarized in the following paragraphs.

5.8.2.2.1. Display Format Data

The UTS 400 display format is controlled by firmware (residing in ROM) in conjunction with display hardware. Screens are formatted in either 64 or 80 displayable characters (bytes) per line and in 12, 16, or 24 lines. The remaining bytes, to a maximum of 96 bytes per line, are FCC positions, one of which is an end-of-line indicator.

A display memory consists of 4K bytes, each byte being 9 bits wide. Eight of the bits are data, and one (bit 7) is an FCC indicator. The FCC indicator is tested by the firmware to locate FCCs. In most cases, information in display memory is in ASCII code, which requires only seven bits per character. These seven bits are located in bit positions 6 through 0 of a byte in display memory. Should bit 8 be required, it must be accessed via the scope status port (SCSPRT). An IN instruction is executed to retrieve the bit (see 5.8.3.2). Writing bit 8 is performed similarly, using an OUT instruction for the scope status port.

The user program may establish FCCs in any position of display memory as desired. However, user programs must avoid exceeding the total of 16 FCCs per line and must also avoid destroying the null used as a line delimiter.

5.8.2.2.2. Display Pointers and Flags

Since the UTS 400 screen formats are variable, user programs should test specific information locations to:

- identify the screen size,
- control the cursor position through the cursor address features, and
- change the display memory without firmware assistance.

Display pointers and flags used to perform this testing are:

BGCURS
BGNXTL
CRTUSR
DCFLAG
FDBDCT
HOMATT
LINES
LSTATT
MXCURS
NUMCHR

The majority of the locations listed are used by user programs for information only. The BGNXTL pointer may, however, be modified by a user program as appropriate when the contents of the CURSOR flag are changed so the cursor is repositioned to another line group. When the contents of the CURSOR flag are changed, bit 6 of the CRTCTL flag must be a binary 0 if the cursor is to be displayed on the screen. In addition, the address placed in the CURSOR flag must correspond to a displayable position; otherwise, the cursor is lost.

A user program may lock the keyboard by setting bit 7 of the CRTCTL flag. However, interrupts must be disabled during this operation.

The CRTCTL flag may be referenced and changed by both interrupt and noninterrupt code. Therefore, any changes must be performed with interrupts disabled. Since independent bits of the CRTCTL flag control various operations, the bits are normally changed by a sequence similar to the following:

```
DI                          Disable interrupts
LDA         CRTCTL          Read existing value
ORI         0200            Set bit 7
STA         CRTCTL          Set updated value
EI                          Enable interrupts
```

This sequence lights the WAIT indicator and locks the keyboard. The keyboard may be unlocked by clearing bit 7.

The poll and audible alarm bits of the CRTCTL flag must be reset for each occurrence; the indicator lights and/or the alarm sounds only once for each write to the flag with the appropriate bits set to binary 1's. Since a write to this flag for the purpose of turning other bits on or off would cause the POLL indicator and audible alarm to respond to the state of the other bits, it is important to leave these bits in the binary 0 state. A sequence similar to the following should be used to set the poll or alarm bit:

```
LXI         H, CRTCTL       Get CRTCTL address
DI                          Disable interrupts
MOV         A, M            Save existing value
MVI         M, 01           Sound alarm
MOV         M, A            Restore existing value
EI                          Enable interrupts
```

The leftmost byte of the CRTUSR flag must be set to a binary 1 each time the user program changes display memory without using firmware services. This should be done without disturbing the other bits of CRTUSR. The user program should not rely on this byte being set or cleared upon entry.

The control page may be scanned by referencing the locations beginning at the DCPAGE flag, which is the first character of the control page, and continuing for 192 bytes, which corresponds to the amount of memory necessary to accommodate the two control page lines.

5.8.2.2.3. Communications Pointers and Flags

The following communications pointers and flags should be read for information only and should not be changed by user programs:

BELFLG
COMDID
OWNDID
OWNSID
RIDFLD

The remaining communications flags and pointers are used to store communications-line data input in user RAM or to output data from user RAM to the communications line.

To store communications input data, the user program places the user memory address and buffer length in the COMIN and COMINL locations, respectively, and sets bit 7 of the COMNOC flag to activate the request to receive input. COMNOC flag bit 7 is reset after each transmission and therefore must be set each time a user program requests communications input to user RAM.

Since all text inputs to the user program are acknowledged (with an ACK response), a subsequent message for the same station may be received before processing is completed for the current message. To avoid conflicts arising from this situation, the user program in the host processor may wait until a text or attention key message is received before sending a subsequent message.

Setting bit 0 of the COMNOC flag indicates a communications output from user RAM. The user memory address and buffer length are first loaded into the COMOT and COMOTL locations, respectively. Then the user program must:

- establish the transmission type in the TYPE field of the control page of the desired station,
- load the transmit code (octal 0160) in register B,
- set bit 0 of the COMNOC flag to a binary 1, and
- return to the firmware at the FUNCFM exit.

As in the case of communications input, the command bit in the COMNOC flag is reset after each output transmission, and therefore bit 0 must be set each time a transmission is to be made from user RAM. Both bit 0 and bit 7 may be set simultaneously to allow output and a subsequent input using user memory.

5.8.2.2.4. Peripheral Pointers and Flags

NOTE:

All references to peripherals in the following discussion pertain to share-type peripherals.

User programs may access and modify information in peripheral pointers and flags in much the same manner as that described for the communications pointers and flags (5.8.2.2.3). However, peripheral operations are controlled through the control page fields; that is, the transfer, print, backward-one-block, report-address, search, and status functions are established by means of the control page.

Before modifying peripheral pointers or flags, the user program should ensure that the control page is not on the screen. If it is, an operator may be in the process of changing values or directing new operations. A test of DCFLAG can be made to determine the location of the control page. The control page can be returned to storage by loading the key code for the control page in register B and exiting at the FUNCFM exit.

Data can be transferred between peripherals and user RAM rather than display memory. Control of such operations is based on the value in the PERNOC flag. If the appropriate flag bit is set, the transfers to user RAM are enabled. Further, unless the PERNOC flag is set for input or output, the values in the PERIN, PERINL, PEROT, PEROTL, and UDISKC locations have no meaning.

An important aspect of peripheral information usage is that of interfacing the diskette. Normally, the first byte of every sector on disk is reserved for sequencing of sectors within a block from the screen. This format may be modified, however, by use of the UDISKC flag. Values can be placed in the UDISKC flag to specify that read and write operations will take place to and from one sector and that all 128 bytes in a sector will be used (see 5.5.2.4.15). In addition, the UDISKC flag may be used to write a control sector on the disk.

5.8.2.2.5. General Pointers and Flags

User program use of this information, which consists largely of indications concerning the condition of the system, is evident from the information descriptions in 5.5.2. However, the information in the LOADFL, DSPNDX, and USERLC locations is of major importance in implementing user programs and therefore bears further discussion.

The LOADFL flag is used by the firmware to determine the existence of a user program. A binary 0 in this flag indicates that there are no user programs in the UTS 400, and the firmware bypasses all user program entry points. Further, execution of any user programs in the UTS 400 may be terminated by storing a binary 0 in this flag.

By storing an octal value of 0252 (hex AA) in the LOADFL flag, one user program may inhibit loading of another user program. An important application of this feature is that of inhibiting the loading of a user program while another is being executed. User programs should clear this condition when its use is no longer required. However, the loading lockout can be cleared by an initialization process using the TEST switch.

The DSPNDX location contains a value that defines the display to be used during execution of a user program. User programs may change the display by modifying the contents of the DSPNDX location. A change in the DSPNDX value requires a corresponding change to the value in the device port (SCXPRT), since these two are used in combination to select the display memory.

Because of the way the DSPNDX and SCXPRT values are used by the firmware, the following sequence should be observed in changing the display memory selection (that is, placing a new index value in the register):

```
STA         DSPNDX        Set dispatcher index

OUT         SCXPRT        Select display memory (see 5.8.3.3)
```

The USERLC flag defines a set of locations set aside for user programs to provide a mechanism for reentrant processing. These locations are resident in display memory and, as such, are duplicated in all display memories and the screen bypass memory. Access to the USERLC locations is made for the display memory defined by the DSPNDX/SCXPRT value.

5.8.3. Device Port Usage

Device ports, as defined in 5.6, are used as the link between the firmware and the hardware of the UTS 400. Although other ports are provided in a UTS 400, only those described in 5.6 may be used by user programs. Control of these ports by user programs is enabled by means of IN and OUT instructions.

5.8.3.1. RAM Boundary Port

The RAM boundary port (RABPRT) is used to separate program instructions from data in user RAM. All locations below the value set in the RAM boundary port are treated as ROM, and an attempt to write to any of these locations will not change the content of the locations. Locations equal to or greater than the value in the RAM boundary port are treated as RAM.

When a user program is loaded, the RAM boundary port is set to an octal value of 0240 (hex A0). This value corresponds to the upper eight bits of the start of user RAM (octal location 0120000, or hex A000). If a user program has no need for the extra bit in any of user RAM, this value need not be reset. However, if the extra bit is needed, the RAM boundary port must be set to a value that corresponds to the lower boundary of any memory location in which the extra bit is to be accessed or set. As a result, a user program must separate instructions and data if the extra bit is to be used.

The following sequence illustrates the method of obtaining all nine bits:

```
LDA      Memory      Read memory

MOV      X, A        Save byte in a register

IN       RASPRT      Get bit 9 from RAM status port (see 5.8.3.2)
```

5.8.3.2. Scope Status and RAM Status Ports

These ports are used to access the extra bit in the appropriate memory. The scope status port (SCSPRT) is used to access the bit in display memory, while the RAM status port (RASPRT) is used to access the bit in user RAM, under control of the RAM boundary port.

The following sequence illustrates the use of the ports in reading memory when the extra bit is used.

```
LDA      Memory      Read memory

MOV      X, A        Store the bits

IN       SCSPRT      Get extra bit
         or
         RASPRT
```

This sequence reads a memory location and then reads the extra bit. Any number of instructions can be located between the memory read and the port read. The result will be the same as long as none of the intervening instructions read memory again, which sets the port.

The following example sequences illustrate the use of the scope status or RAM status port in writing memory when the extra bit is used.

Example 1:

```
MVI      A, 0210

OUT      SCSPRT      Set extra bit to binary 1
         or
         RASPRT

MOV      M, B        Write to memory. The extra bit
                     is set, and register pair H, L is set.
```

Example 2:

```
LDA      Memory      Read nine bits of data: eight in register A
                     and one in scope or RAM status port.

STA      Memory+1    Write the nine bits

STA      Memory+2    Write the nine bits
```

The write memory sequence sets the extra bit in memory. Once set, the extra bit maintains its value until set again by a write to this port or by a read from memory.

5.8.3.3. Scope Index Port

The scope index port (SCXPRT) is used in conjunction with the DSPNDX pointer as an index mechanism for selection of display memories. The values of DSPNDX and SCXPRT must be the same and maintained thusly by the user program. (Refer to 5.5.2 for a description and illustration of this mechanism.)

Each bit except bit 7 of the scope index port represents a display memory. A memory is selected by a binary 0 in a bit position; therefore, only one bit should be a binary 0.

Selection of a display memory may be performed by the following sequence:

```
        MVI     A, 0376     Select primary station

        STA     DSPNDX      Set dispatcher index

        OUT     SCXPRT      Select display memory
```

All display memories are strapped so that consecutive bits are used for the memories configured. Bit 0 is normally used to select the master or primary slave display. The SYSNDX flag (5.5.2.5.8) contains an indication of the last operable display memory in the system.

The following sequence is an example of a control loop that permits the user program to select each of the display memories configured and to perform the same set of instructions for the selected display memories.

```
        LDA     SYSNDX      Get system index byte

        MOV     B, A        Store the byte

LOOP

        MOV     A, B

        RRC                 Shift to select next display memory

        JNC     END         Go to END if all display memories have been selected

        MOV     B, A        Save new selection

        STA     DSPNDX

        OUT     SCXPRT      Select the display memory

         .          . )
         .          . }     Intervening instructions for the selected display
         .          . )     memory. Register B should not be changed.

        JMP     LOOP        Return to LOOP and select next display memory

END

         .          . )
         .          . }     Instruction(s) following LOOP
         .          . )
```

5.9. USER PROGRAM LOADING

User programs designed for execution on a UTS 400 are assembled and generated on a host processor. Loading of the user programs into UTS 400 memory may be performed by remote or local mechanisms.

A remote user program load into UTS 400 memory is accomplished via block transfers from a host processor over a communications line. Local loads are also performed via block transfers, but the user programs are derived from a locally attached UTS 400 storage device. However, the locally stored user programs are originally derived from a host processor block transfer over a communications line; that is, a host processor first transmits the user programs as data through a UTS 400 to a local storage device.

The UTS 400 maintains firmware control until a complete program is loaded regardless of the loading mechanism. Any user program being executed at the time a load is initiated will be terminated if the LOADFL flag is not set to an octal value of 0252 (hex AA). When the load is completed, control is passed to the user program.

In addition to loading the UTS 400, a host processor can also recall a user program from the UTS 400. The formats for remote and local loading and for recalling user programs are described in the following paragraphs.

5.9.1. Remote Load Format

Remote loading of user programs into UTS 400 memory is performed via block transfers from a host processor over a communications line. The number of block transfers (one or more) required to load a user program depends on the user program size and the transmission block size.

The block format for a remote load is defined in Figure 5–6. All bytes of the blocks contain a parity bit in the most significant bit location. A hex value of 30 (octal 60) is added to each pair of four bits in each byte of start addresses A and B and to each byte in the user program. Control code 0210 is used to direct a user program load into UTS 400 RAM. The most significant four bits of start addresses A and B are transferred first. Each program byte is transferred as two bytes (with the hex 30 value added to each 4-bit pair), and with the most significant four bits of the original program byte transferred first. ←

Start addresses A and B contain an identical address code defining the starting memory address for the user program. Start address B is a redundant parameter included to ensure that the program is loaded into RAM at the proper address.

During loading of the program, the only control characters that are detected are NUL, STX, SOH, and ETX. Any other control characters are used as part of the load data. Detection of an ETX terminates the program load block.

When a block containing an ETX is received immediately following start address B, program control is passed to the absolute address in the block if the load was successful. This address is defined as the transfer address.

Each block is checked using the parity bits and the BCC. Should a transmission error occur, the block will not be acknowledged and the block must be retransmitted, as defined in Section 3, paragraph 3.7.

A second form of checking, which involves a check of the block format, is also performed. A message is returned to the host processor following receipt of the transfer address indicating either a successful load sequence or an error during the sequence. The format of each message type is shown in Figure 5–7.

The two messages are similar except for the control code. A value of 9000 is returned for a successful load sequence, while a value of 904* is returned for a format error. The last character (*) defines the type of error encountered. The format of this byte is shown as follows:

Bit	6	5	4	3	2	1	0
	1	X	X	X	X	X	X

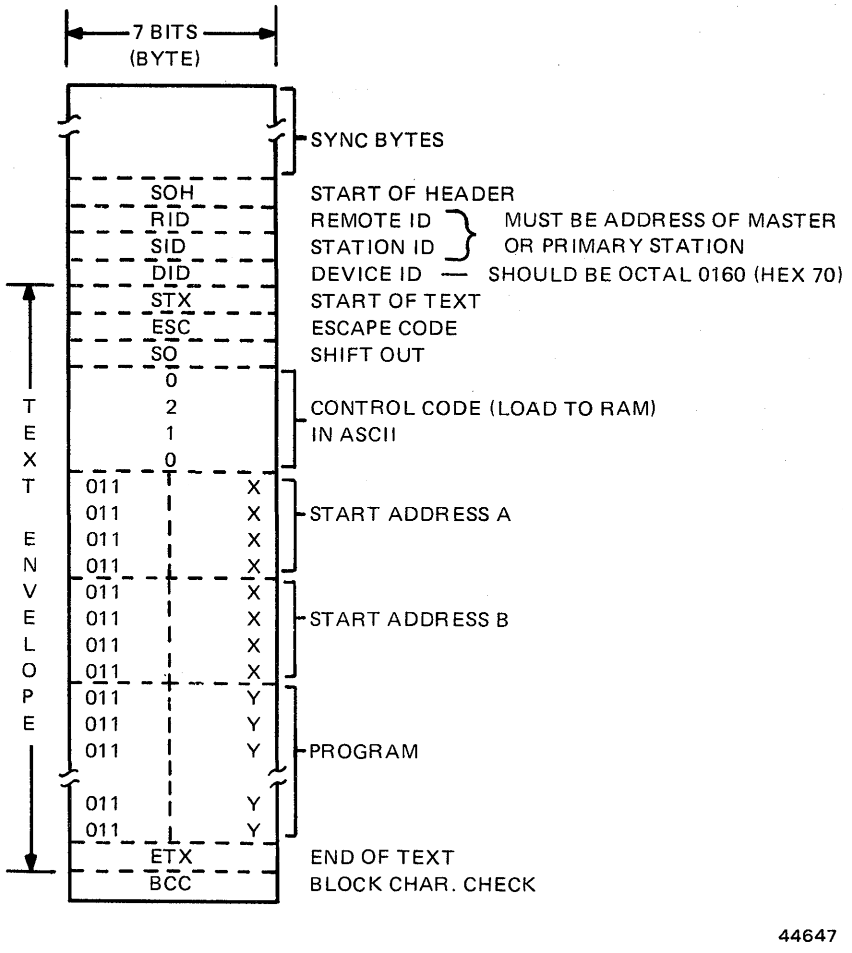


Figure 5–6. Communications Load Block Format to UTS 400

A binary 1 in a bit location indicates an error. Multiple 1 bits are returned if multiple errors are encountered. The error types and corresponding bit locations are as follows:

Bit Number	Error Type
0	Start address A not equal to start address B.
1	Start address was out of range, block not loaded.
2	Block overflowed available memory.
3	Illegal control code encountered in a block, and block not loaded.
4	Load sequence not directed to a master or primary station, and block not loaded.
5	Load sequence not honored because LOADFL flag was set to octal value 0252 (hex AA) to disallow a load.

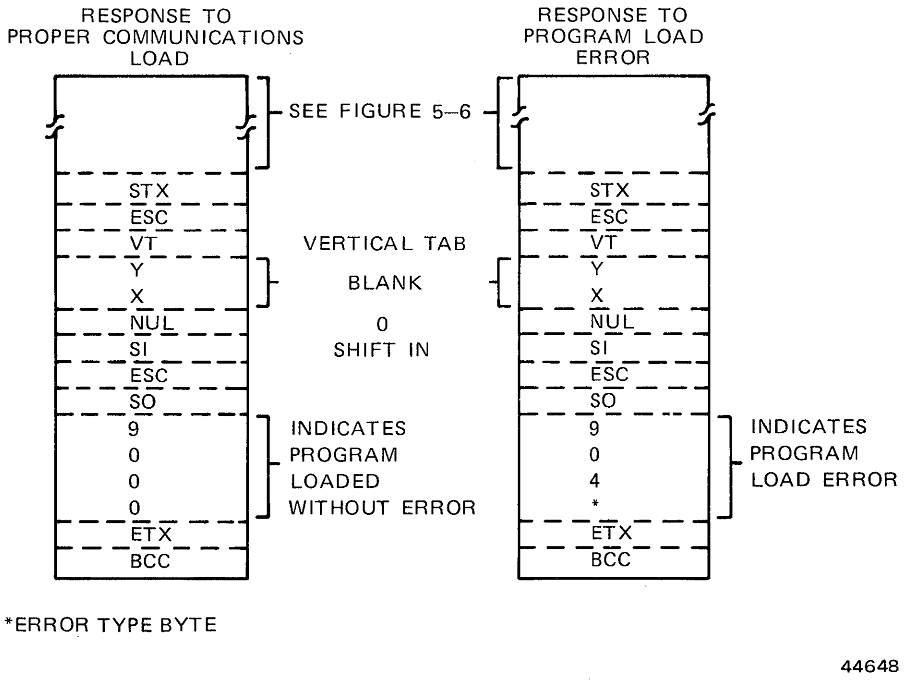

RESPONSE TO PROPER COMMUNICATIONS LOAD
RESPONSE TO PROGRAM LOAD ERROR
SEE FIGURE 5–6
STX
ESC
VT
Y
X
NUL
SI
ESC
SO
9
0
0
0
ETX
BCC
VERTICAL TAB
BLANK
0
SHIFT IN
INDICATES PROGRAM LOADED WITHOUT ERROR
STX
ESC
VT
Y
X
NUL
SI
ESC
SO
9
0
4
*
ETX
BCC
INDICATES PROGRAM LOAD ERROR
*ERROR TYPE BYTE
44648

Figure 5–7. UTS 400 Block Format Response to Communications Load

5.9.2. Upline Recall Request

Using an upline recall request message, a host processor can request that a copy of a user program in UTS 400 memory be returned. The text envelope portion of this recall message is shown in Figure 5–8. This message may be sent independently or as part of a block containing other messages. In either case, the information in the text envelope is the same.

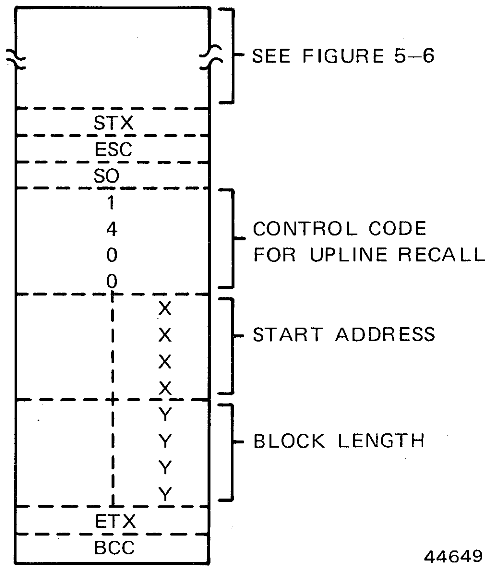


Figure 5–8. Host Processor Communications Request Format for Upline Recall

Control code 1400 is the upline recall request. The start address and block length of the user program to be returned are defined in bytes following the control code.

As with the formats described in 5.9.1, a hex value of 30 is added to each of the 4-bit pairs of each byte in the start address and block length. The most significant four bits are transferred first.

5.9.3. Upline Program Sending

In response to an upline recall request, the UTS 400 returns the requested user program to the host processor. The block format for returning the user program is shown in Figure 5–9.

Control code ;003 indicates that a program is being returned. Both the start address and block length of the program precede the requested program.

The start address and block length correspond to the parameters sent to the UTS 400 as part of the upline recall request (5.9.2). However, should the upline recall request include a start address for a portion of memory not included in the user RAM area, the block length returned is set to 0000. Additionally, if the block length in the upline recall request extends beyond the available user memory area, the block length returned is set to the number of memory locations transferred.

As with the formats described in 5.9.1, a hex value of 30 is added to each of the 4-bit pairs of each byte of the start address, block length, and program. The most significant four bits are transferred first.

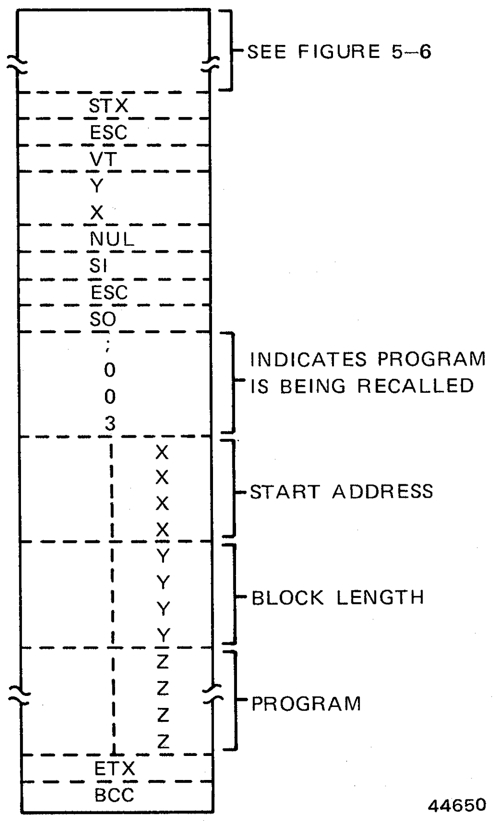


Figure 5–9. UTS 400 Block Format Response for a Host Processor Upline Recall

5.9.4. Offline Program Load

A host processor can direct a user program (as data) to a storage device attached to a UTS 400. In this case, the UTS 400 is transparent to the user program loading and stores the user program on an attached storage device in the same manner as any other message directed to a UTS 400 peripheral. The block format for a user program transfer to a peripheral is shown in Figure 5–10.

The DID in the message is the identifier for the attached storage device. In most respects, the host processor block transfers are in the same format as that used for remote load directly to the UTS 400 memory (5.9.1). The major differences, other than the DID, are the inclusion of device control characters (DC3 DC3), replacement of the control code with a program label field, and inclusion of the print transparent command.

Loading of the user program into UTS 400 memory from the attached storage device is initiated by pressing the LOAD PROGRAM key on the UTS 400 keyboard. The format of the user program from the attached storage device is the same as that shown in Figure 5–10 except that all bytes preceding the DC3 DC3 bytes and those following the program are not included.

Once initiated, the program load continues, block after block, until the transfer address is detected. This transfer address is a block with a start address but no program content. At this point, program control is passed to the user program at the absolute address defined by the transfer address.

If a format error occurs during an offline load operation, the UTS 400 firmware retains control, the NOPR-X status is placed in the ADR field of the control page, and an alarm sounds. The character in the X position of the NOPR-X status defines the type of format error. The bit format used for the X character is the same as that described for the error type byte in 5.9.1.

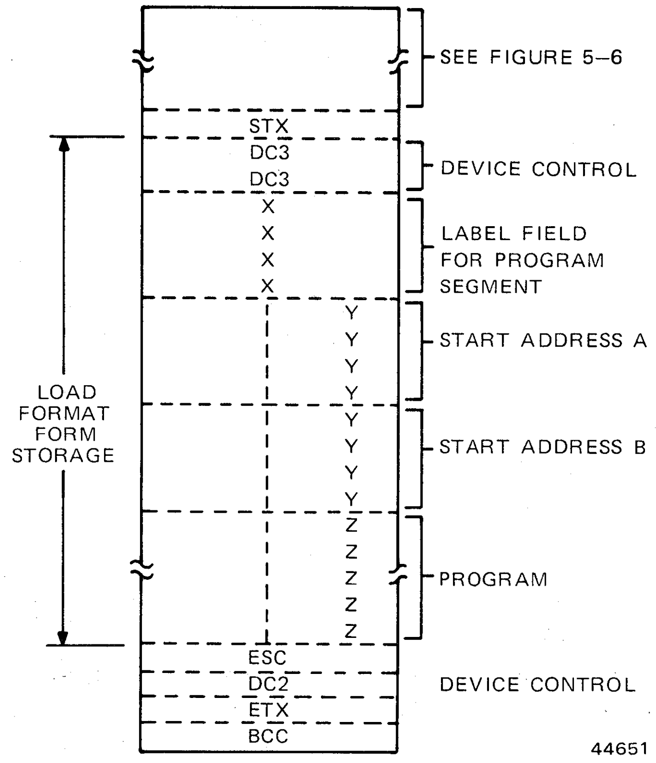


Figure 5–10. Communications Load Block Format to UTS 400 for Offline Program Load

5.10. PROGRAMMING CONFLICTS

Certain instructions in the UTS 400 instruction set should be used with extreme care, and some should not be used at all. Caution should be exercised in using the following instructions:

Mnemonic	Machine Code (hex)	Definition
IN	DB	Input
OUT	D3	Output
EI	FB	Enable interrupts
DI	F3	Disable interrupts

The following instructions, if sent downline to the UTS 400, could create conflicts between user programs and the firmware. Therefore, these instructions should not be used in writing user programs:

Mnemonic	Machine Code (hex)	Definition
DAD SP	39	Add stack pointer contents to register pair H, L
DCX SP	3B	Decrement stack pointer
HLT	76	Halt
INX SP	33	Increment the stack pointer
LXI SP	31	Load immediate stack pointer
RST	Binary 11XXX111	Restart program at octal address 0X0
SPHL	F9	Transfer contents of H, L register pair to stack pointer

The suggested maximum execution time for user programs is 100 milliseconds when the UTS 400 is operating online to a host processor or when peripheral operations are active. User programs may contain control for longer periods of time if no other operations require service. (This time restriction is not a consideration when the UTS 400 is being operated in a stand-alone mode.)

When a user program has control, input data from a communications line or a peripheral is queued for service. If the user program maintains control for periods longer than the suggested maximum time, data may be lost because of queue overflows.

5.11. USER PROGRAMMABILITY CONSIDERATIONS FOR FIRMWARE REVISION LEVEL G

This paragraph defines the additional areas of firmware control necessary to program a UTS 400 incorporating firmware revision level G and later revision levels.

NOTE:

The revision level of the firmware in a given station is presented as the last character in the second line of the control page, as shown in 2.4.

5.11.1. Overview of Firmware Revision Level G

Level G firmware increases the UTS 400 throughput by adding several tables and pointers related to screen management. These tables and pointers must be updated when an entry is made into screen memory or when the cursor is moved. The user-changed flag (CRTUSR), described in 5.5.2.2.4, provides an indication to the firmware that the screen has been modified by a user program. Control of the CRTUSR flag must be maintained or erroneous screen format may result from firmware action. CRTUSR control considerations are covered in 5.11.2.

The enhanced performance resulting from firmware level G also makes additional display memory available for user programs, as discussed in 5.11.4.

5.11.2. CRTUSR Flag Control

As stated in 5.5.2.2.4, the CRTUSR flag byte must be set by the user program whenever the user program modifies display memory. Execution of user programs under firmware level G requires that the user program set the CRTUSR flag in the following circumstances:

- When the cursor is moved from one location on the screen to another location, even within the same line
- When any character or FCC in display memory is changed

The CRTUSR flag has to be set only once before the user program exits to the firmware, no matter how many times the user has changed the screen since control was received from the firmware. However, the flag must be set each time the user receives control if the screen is changed by the user program before an exit is made to the firmware.

5.11.3. USRCTL Byte

This byte was originally labeled CRTRSV. ←

The contents of this byte, located at CRTUSR+1 in each display memory, indicate to the user program that the operator has pressed one of the following keys:

	Code in USRCTL Byte			Code in USRCTL Byte	
Key	Octal	Hex	Key	Octal	Hex
Keyboard Unlock	0177	7F	Attention Key F2	0107	47
Message Waiting	0007	07	Attention Key F3	0127	57
Attention Key F1	0067	37	Attention Key F4	0147	67

The associated code is placed in the USRCTL byte when the key is pressed by the operator. If more than one of the keys is pressed before the user program receives control, the code of the last key pressed will appear in the USRCTL byte. The firmware never clears this byte (except at POC time); therefore, the user program should clear it, when necessary, to remove a previous code already acted upon.

5.11.4. Additional CRT Memory Locations Available to User Programs

Level G firmware provides 32 additional bytes of display memory for user programs, beginning at octal location 035240 (hex 3AA0). These 32 bytes are in addition to the 8 bytes of user memory beginning at octal location 035063 (hex 3A33).

It should be noted that programs written to use the extra display memory available in firmware level G will not execute on firmware below that level. The reverse is not true, however; programs using the eight bytes of display memory available in lower levels of firmware can be executed with level G, providing the CRTUSR flag is controlled as defined previously.

Appendix A. Line Protocol Rules

The line protocol rules listed in this appendix are discussed in detail in Section 3. The parenthetical paragraph reference located after each rule number indicates the paragraph in which a complete discussion of that rule can be found.

A.1. UTS 400 RULES

UTS 400 Rule 1 (refer to 3.5.1):

The station responds only to error-free polls.

UTS 400 Rule 2 (refer to 3.5.1):

The station expects an acknowledgment to any message it sends in response to a poll except the no-traffic response.

UTS 400 Rule 3 (refer to 3.5.1):

A station will not send two consecutive text messages. Thus, a station does not respond with a text message to a poll that includes the acknowledgment to the previous text message.

UTS 400 Rule 4 (refer to 3.5.1):

The station acknowledges in its next poll response any error-free host processor message containing an STX (text message, message wait command, or disconnection command).

UTS 400 Rule 5 (refer to 3.7.1):

If a station that is owed an acknowledgment does not receive an acknowledgment in the next good poll that it recognizes, the station sends a reply request (DLE ENQ) message.

UTS 400 Rule 6 (refer to 3.7.1):

A station will not send an acknowledgment (ACK or WABT) with a reply request. A station having a reply request can be passed an acknowledgment because of the multiplexer function. However, that acknowledgment will not be reported until the reply request condition is satisfied.

UTS 400 Rule 7 (refer to 3.8.4):

The specific DID in any host processor message causes a selection attempt only if the message is error free.

UTS 400 Rule 8 (refer to 3.8.4):

The UTS 400 will respond to a selection poll or to the first poll following a TEXT/SD, TEXT/SD/PI, or TEXT/PI message with a peripheral status response. These peripheral status responses are listed in Table A–1.

Table A-1. UTS 400 Peripheral Status Responses to Host Processor Messages

Peripheral Condition	Selection Poll From Host Processor	TEXT/SD From Host Processor	TEXT/SD/PI From Host Processor	TEXT/PI From Host Processor (after previous selection)
No peripheral interface support within UTS 400	No traffic	DLE 1	DLE 1	DLE 1
Peripheral response 1 (ready)	DLE >	DLE 1 DLE >	DLE ? or DLE 1 DLE ;*	DLE ? or DLE 1 DLE ;*
Peripheral response 2	DLE <	DLE 1 DLE <	DLE 1 DLE < or DLE ?**	DLE ?
Peripheral response 3	DLE :	DLE 1 DLE :	DLE 1 DLE : or DLE ?**	DLE ?
No peripheral response	DLE =	DLE 1 DLE =	DLE 1 DLE =	DLE ?
Peripheral selection delayed (another station using the peripheral interface)	DLE 5	DLE 1 DLE 5	DLE 1 DLE 4	DLE 1 DLE 4

*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 host processor.

**The response depends on the peripheral. An acknowledgment plus error status implies the peripheral initiation did not occur. The busy status implies the initiation was attempted. If the peripheral initiation was successful, the THRU status will subsequently be reported; if unsuccessful, a sustained busy will result. For example, if an end-of-tape condition exists on a tape cassette, an attempt to write would result in the acknowledgment plus end-of-tape status (peripheral status 2) response, an attempt to read would result in a sustained busy status, and an attempt to rewind would be successful (result in a THRU status).

UTS 400 Rule 9 (refer to 3.8.4):

The UTS 400 will not respond to a selection poll or to a TEXT/SD/PI message when an offline peripheral buffering operation is in progress. This condition appears to the host processor as if an error occurred on the poll or response. The normal no-response error recovery should be invoked.

UTS 400 Rule 10 (refer to 3.8.4):

A selection poll requiring access to a peripheral interface already in use by another station will solicit the peripheral-selection-delayed (DLE 5) response. The station for which the selection is to be performed is added to the PAQ. When that station reaches the top of the PAQ, the selection attempt is made and the next general poll or specific poll response from that station will be DLE > (peripheral ready), DLE <, DLE :, or DLE = (no response from the peripheral). The interpretations of DLE < and DLE : depend on the peripheral in use.

UTS 400 Rule 11 (refer to 3.8.4):

A TEXT/PI message or a TEXT/SD/PI message requires access to the peripheral interface. If the peripheral interface is already in use, the station response is ACK plus message queued (DLE 1 DLE 4). The station response to subsequent general polls is no traffic, and the response to subsequent specific polls is message queued (DLE 4) until the message reaches the top of the PAQ. The station then responds to subsequent specific polls with a WABT (DLE ?) as long as the peripheral operation is in progress. The station response continues to be no traffic to general polls. A THRU (DLE ;) status is sent as a poll response upon completion of the peripheral operation.

UTS 400 Rule 12 (refer to 3.8.4):

Upon completion of a data transfer to or from a peripheral, the UTS 400 sends the THRU (DLE ;) status as a poll response. When automatic peripheral retry is enabled in a station, the THRU response means the peripheral operation was completed successfully. Without automatic peripheral retry, the data transfer may have been completed successfully, but the operation of certain devices (specifically, the tape cassette system) may not be complete when the THRU response is returned.

UTS 400 Rule 13 (refer to 3.9.1):

A station will not respond with a message-waiting or program-attention-key code message to a poll containing an acknowledgment if that station is owed an acknowledgment.

UTS 400 Rule 14 (refer to 3.9.1):

If a UTS 400 station or cluster has more than one response to send, the following priority sequence will be observed:

1. Any station or peripheral status responses are sent first. These responses include:
   a. Message-queued response (DLE 4) due to receipt of a TEXT/PI or TEXT/SD/PI message when other messages were already in the PAQ
   b. Peripheral-selection-delayed response (DLE 5) due to receipt of a selection poll or a TEXT/SD message when other messages were already in the PAQ
   c. Peripheral-operation-THRU response (DLE ;) due to completion of the peripheral operation
   d. Peripheral status response 1 (DLE >) due to receipt of a TEXT/SD message or a selection poll when no other messages were in the PAQ
   e. Peripheral status response 2 (DLE <), 3 (DLE :), or 4 (DLE =) due to receipt of a selection poll, a TEXT/SD message, or a TEXT/SD/PI message when no other messages were in the PAQ
2. If no priority 1 conditions exist, then any text messages resulting from the XMIT key being pressed or from a host processor transmit command are sent.
3. If no priority 2 conditions exist, then any message-waiting or program-attention key code messages are sent.

A.2. HOST PROCESSOR RULES

Host Processor Rule 1 (refer to 3.5.1):

Upon sending a text message to a station, the host processor must poll to verify proper receipt of the text. This poll must occur before any other text is sent to that poll group.

Host Processor Rule 2 (refer to 3.5.1):

The host processor must send a poll with acknowledgment to a station that has sent an acknowledgeable response before the host processor can send a text message to that station.

Host Processor Rule 3 (refer to 3.6.1):

When using general polls, the host processor must expect an acknowledgment (ACK or WABT) from one station to be included with a response from another station in the poll group. The multiplexer function allows an acknowledgment to be passed from one station to another station that has a traffic response pending.

Host Processor Rule 4 (refer to 3.6.1):

When using general polls, the host processor must expect successive traffic responses from stations within a poll group. However, UTS 400 rule 3 still applies. The multiplexer function allows a station with a message pending to send its response to any poll whose address it recognizes.

Host Processor Rule 5 (refer to 3.6.1):

The host processor can send text to any station in the poll group that is not owed an acknowledgment, provided the last response from the poll group was not one of the following:

- ACK only (SOH RID SID DID DLE 1 ETX BCC)
- WABT only (SOH RID SID DID DLE ? ETX BCC)
- ACK plus text available (SOH RID SID DID DLE 1 DLE 0 ETX BCC)
- WABT plus text available (SOH RID SID DID DLE ? DLE 0 ETX BCC)

Host Processor Rule 6 (refer to 3.6.1):

When the host processor owes an acknowledgment to a station in a poll group, the host processor may send a specific poll to the poll group only if the specific poll is addressed to the station that is owed the acknowledgment.

Host Processor Rule 7 (refer to 3.6.1):

The host processor must not allow a given poll group to owe more than one acknowledgment to the host processor at any time.

Host Processor Rule 8 (refer to 3.7.1):

The host processor must treat any error in a received message as a no-response condition and must repeat the poll that preceded the no-response condition. However, any acknowledgment included with the original poll must be eliminated and a general DID must be used when the poll is repeated. If the no-response condition results from a retransmission request (DLE NAK), the host processor must repeat the poll that created the reply-request response.

Host Processor Rule 9 (refer to 3.7.1):

The host processor response to a reply request (DLE ENQ) from a station must be a retransmission request (DLE NAK) if the last message correctly received from the station sending the reply request was one not containing text data. The retransmission request has the same specific RID and specific SID address as that contained in the reply request. The retransmission request must contain a general DID.

Host Processor Rule 10 (refer to 3.7.1):

The host processor response to a reply request (DLE ENQ) from a station must be a poll-with-acknowledgment message if the last message correctly received from the station sending the reply request was a message whose traffic was text. The acknowledgment is for the message that preceded the reply request and not for the reply request message itself.

Host Processor Rule 11 (refer to 3.7.1):

The host processor response to a reply request (DLE ENQ) from a station must be a retransmission request (DLE NAK) if the station sending the reply request is other than one to which an acknowledgment has just been sent.

Host Processor Rule 12 (refer to 3.7.1):

If the response from a retransmission request is identical to the UTS 400 response sent just previous to the reply request and is from the same station, then the response to the retransmission request can be ignored except for sending the acknowledgment that the station expects. However, the same UTS 400 response is to be ignored only once.

Host Processor Rule 13 (refer to 3.8.4):

If a no-response condition exists on a selection poll, then host processor rule 6 (which requires sending of a specific poll) must be followed. If the response to this specific poll is a reply request, then a retransmission request must be sent. If the response is no traffic, the selection poll must be resent.

Host Processor Rule 14 (refer to 3.8.4):

A selection poll can contain an acknowledgment only if the selection is being performed on the station that is owed the acknowledgment.

Host Processor Rule 15 (refer to 3.8.4):

The host processor must maintain peripheral-operation timers at the station level and/or at the UTS 400 cluster level to provide an indication of excessive wait time or of a sustained busy condition. These timers must take into consideration the amount of time a station is on the PAQ prior to performance of the operation (that is, the wait time) and the amount of time required to actually perform the operation. Wait time depends on the number of stations in the cluster, the number of host-processor- and operator-initiated operations already in the PAQ, the type of operations being performed, and whether or not automatic peripheral retry is in effect. The amount of time required to actually perform the operation depends on the type of operations being performed and whether or not automatic peripheral retry is in effect.

The station-level timer should be initiated for:

- A delayed status response to a selection poll

- An ACK plus delayed status response to the poll following a TEXT/SD message

Host Processor Rule 15 (cont)

- An ACK plus message-queued response to the poll following a TEXT/SD/PI message
- A WABT response to the poll following a TEXT/PI or TEXT/SD/PI message

When general polls are being used, the station-level peripheral timeouts can be used as a timing mechanism for periodically sending specific polls to determine when the peripheral operation is actually in progress (see UTS 400 rule 11.

The station-level timer should be terminated upon notification of a peripheral completion (the THRU response if the first response was WABT or ACK plus message queued, or the peripheral device status if the first response was delayed status).

A cluster-level timer should be initiated for the same poll responses as those listed for a station-level timer. However, once a cluster-level timer is started, similar poll responses from the cluster will not be used to restart the timer. Upon notification of a peripheral completion (THRU if the first response from the station was WABT or ACK plus message queued, or peripheral device status if the first response was delayed status), the host processor must reinitiate the cluster-level timer if there are other host-initiated operations pending for that cluster.

Appendix B. UTS 400 Codes and Code Sequences

B.1. COMMUNICATIONS CONTROL CHARACTERS

Control characters used by the SPERRY UNIVAC Universal Terminal System 400 are shown in columns 0 and 1 of the chart in Figure B–1. The communications control characters are defined as follows:

B.1.1. Null (NUL)

NUL is an all-zeroes character that may be used 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. The NUL character may be arbitrarily added at any point in the transmission except:
  - In a control sequence following DLE
  - Between ETX and the next following BCC (3.4.3.2)
  - Between the address identifiers in a heading

B.1.2. Start of Heading (SOH)

The SOH character is used at the beginning of a sequence of characters that constitutes a machine-sensible address or routing information.

Usage:

- SOH marks the start of a message heading.

	CONTROL CHARACTERS		DATA CHARACTERS						HEXADECIMAL				
			64 CHARACTERS (UPPERCASE)				32 CHARACTERS (LOWERCASE)						
	0	1	2	3	4	5	6	7	← LEFT DIGIT				RIGHT DIGIT
MS	0	0	0	0	1	1	1	1	7	BINARY BIT NUMBER			
	0	0	1	1	0	0	1	1	6				
	0	1	0	1	0	1	0	1	5 4	3	2	1	
	NUL 000	**DLE** 020	**SP** 040	**0** 060	**@** 100	**P** 120	**`** 140	**P** 160	0	0	0	0	0
	SOH 001	**DC1** 021	**!** 041	**1** 061	**A** 101	**Q** 121	**a** 141	**q** 161	0	0	0	1	1
	STX 002	**DC2** 022	**"** 042	**2** 062	**B** 102	**R** 122	**b** 142	**r** 162	0	0	1	0	2
	ETX 003	**DC3** 023	**#** 043	**3** 063	**C** 103	**S** 123	**c** 143	**s** 163	0	0	1	1	3
	EOT 004	**DC4** 024	**$** 044	**4** 064	**D** 104	**T** 124	**d** 144	**t** 164	0	1	0	0	4
	ENQ 005	**NAK** 025	**%** 045	**5** 065	**E** 105	**U** 125	**e** 145	**u** 165	0	1	0	1	5
	ACK 006	**SYN** 026	**&** 046	**6** 066	**F** 106	**V** 126	**f** 146	**v** 166	0	1	1	0	6
	BEL 007	**ETB** 027	**'** 047	**7** 067	**G** 107	**W** 127	**g** 147	**w** 167	0	1	1	1	7
	BS 010	**CAN** 030	**(** 050	**8** 070	**H** 110	**X** 130	**h** 150	**x** 170	1	0	0	0	8
	HT 011	**EM** 031	**)** 051	**9** 071	**I** 111	**Y** 131	**i** 151	**y** 171	1	0	0	1	9
	LF 012	**SUB** 032	***** 052	**:** 072	**J** 112	**Z** 132	**j** 152	**z** 172	1	0	1	0	A
	VT 013	**ESC** 033	**+** 053	**;** 073	**K** 113	**[** 133	**k** 153	**{** 173	1	0	1	1	B
	FF 014	**FS** 034	**,** 054	**<** 074	**L** 114	**** 134	**l** 154	**¦** 174	1	1	0	0	C
	CR 015	**GS** 035	**–** 055	**=** 075	**M** 115	**]** 135	**m** 155	**}** 175	1	1	0	1	D
	SO 016	**RS** 036	**.** 056	**>** 076	**N** 116	**∧** 136	**n** 156	**~** 176	1	1	1	0	E
	SI 017	**US** 037	**/** 057	**?** 077	**O** 117	**_** 137	**o** 157	**⫽** 177	1	1	1	1	F
			RID			SID		DID		LS			

NOTES:

1. Octal codes appear in the square with each character.

2. General identifier (GID) characters are circled.

3. MS = most significant, LS = least significant.

4. The ⫽ symbol (column 7, octal code 177) is the UTS 400 symbol for the ASCII delete character (DEL).

5. Examples:

   A. For the dollar sign character ($), the octal code is 044, the hexadecimal code is 24, and the binary code is 0 1 0 0 1 0 0.
   7 6 5 4 3 2 1 ← BIT NUMBER

   B. Hexadecimal code 6E is for the lowercase (n) character.
   left digit ↗ ↖ right digit
   The octal code is 156, and the binary code is 1 1 0 1 1 1 0.
   MS ↗ ↖ LS

44620

Figure B–1. UTS 400 Code Chart, Based on ASCII

B.1.3. Start of Text (STX)

The STX character precedes a sequence of characters referred to as text. STX may be used to terminate a sequence of characters (the heading) started with SOH.

Usage:

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

B.1.4. End of Text (ETX)

The ETX character terminates 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.

B.1.5. End of Transmission (EOT)

The EOT character is used to indicate no traffic without acknowledgment in response to a poll; in this context, it is used in a redundant form (EOT EOT) to increase reliability. The EOT character is also used in a DLE sequence to indicate line disconnection (DLE EOT).

Usage:

- Receipt of the EOT 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 DLE EOT sequence indicates that the UTS 400 or host processor is about to disconnect and the recipient should do the same.
- The EOT is never transmitted from a host processor to a terminal except as part of the disconnection sequence (DLE EOT).

B.1.6. Enquiry (ENQ)

The ENQ character is 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.

B.1.7. Synchronous Idle (SYN)

The SYN character is used in 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.
- Conventions applying to use of the SYN character are 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 configuration 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 (3.4.3.2)
    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.

B.1.8. Data Link Escape (DLE)

The DLE character is used to change 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.

B.2. UTS 400 COMMUNICATIONS CONTROL SEQUENCES

The communications control sequences used by the UTS 400 are listed in Table B–1.

Table B–1. UTS 400 Communications Control Sequences

Definition	Code	Paragraph Reference
No traffic	EOT EOT	3.4.2.2
Reply request	DLE ENQ	3.4.2.1
Retransmission request	DLE NAK	3.3.4, 3.4.1.1.
Acknowledgment	DLE 1	3.4.2.3
Text available from station	DLE 0	3.4.2.4.4
Line disconnection	DLE EOT	3.4.2.4.5, 3.10
Busy (WABT)	DLE ?	3.4.2.3
THRU	DLE ;	3.4.2.4.4
Message queued	DLE 4	3.4.2.4.4.
Peripheral device selection delayed	DLE 5	3.4.2.4.4
Power-on confidence test just completed	DLE 6	3.4.2.4.4
Peripheral status 1 (ready)	DLE >	3.4.2.4.3
Peripheral status 2*	DLE <	3.4.2.4.3
Peripheral status 3*	DLE :	3.4.2.4.3
Peripheral status 4 (no response)	DLE =	3.4.2.4.3

*Status meaning depends on peripheral device.

B.3. STATION AND PERIPHERAL DEVICE CONTROL CODES

The station and peripheral device control codes used by the UTS 400 are given in Table B–2.

Table B–2. Station and Peripheral Device Control Codes

Function	Code/Sequence	Par. Ref.	Function	Code/Sequence	Par. Ref.
Cursor positioning	ESC VT Y X SI	4.2.2.1	Transfer changed	ESC E	4.6.1
SOE position	ESC VT Y X NUL SI	4.3.1	Transfer variable	ESC F	4.6.1
Start of entry (SOE)	RS	4.3.1	Transfer all	ESC G	4.6.1
Cursor return (new line)	CR	4.2.2.3	Print form	ESC H	4.6.1
Cursor to home	ESC e	4.2.2.2	Print	DC2	4.6.1
Send cursor address	ESC T	4.5.1	Print transparent	ESC DC2	4.6.1
Erase unprotected data	ESC a	4.4.2.1	Transmit variable	DC1	4.5.6
Erase to end of line	ESC b	4.4.2.3	Transmit all	ESC DC1	4.5.5
Erase to end of field	ESC K	4.4.2.2	Transmit changed	ESC t	4.5.7
Erase display	ESC M	4.4.2.4	Clear changed	ESC u	4.5.8
Erase character (space)	SP	4.4.2.5			
			Call error log	ESC P	4.5.2
Delete in line	ESC c	4.4.3.1	Clear error log	ESC R	4.5.3
Delete in display	ESC C	4.4.3.2			
Delete line	ESC k	4.4.3.3	Initiate confidence test	ESC Q	4.5.4
Insert in line	ESC d	4.4.4.2			
Insert in display	ESC D	4.4.4.3	Blinking start marker	FS	4.4.7.2
Insert line	ESC j	4.4.4.1	Blinking end marker	GS	4.4.7.3
Line duplication	ESC y	4.4.5			
			Lock keyboard	DC4 or ESC DC4	4.5.11
Scan left	ESC g	4.2.2.4			
Scan right	ESC h	4.2.2.4	Shift in	SI	4.4.1.4.1,
Scan down	ESC i	4.2.2.4	Shift out	SO	4.4.1.4.2
Scan up	ESC f	4.2.2.4			
			Line feed	LF	4.6.2.2
Forward tab	HT	4.4.6.3	Form feed	FF	4.6.2.1
Tab stop set	ESC HT	4.4.6.1			
Backward tab	ESC z	4.4.6.4	FCC character sequence	US R C M N	4.4.1.2
			FCC character clear	ESC w	4.4.2.6

B.4. CODE SEQUENCES TO BE AVOIDED

Code sequences to be avoided in transmissions from the host processor to the UTS 400 are given in Table B–3.

Table B–3. Code Sequences to Be Avoided

Sequence	Function
ESC J	Recover auxiliary buffer
ESC I	Release auxiliary buffer
ESC m	Back one block
ESC n	Search
ESC v	FCC reenable
ESC x	FCC locate
ESC N	Auxiliary status
ESC q	Print
ESC r	Transfer
ESC s	FCC generate
ESC p	Transmit
ESC L	Keyboard unlock
ESC l	Report address
ESC O	Hang up

B.5. PROGRAM ATTENTION KEY CODES

The control codes generated by the program attention keys are given in Table B–4. (These codes are discussed in 3.4.2.4.2.)

Table B–4. Program Attention Key Codes

Key Label	ASCII Character	Octal Code	Hexadecimal Code
MSG WAIT	BEL	007	07
F1	7	067	37
F2	G	107	47
F3	W	127	57
F4	g	147	67
F5	Space	040	20
F6	!	041	21
F7	"	042	22
F8	#	043	23
F9	$	044	24
F10	%	045	25
F11	&	046	26
F12	'	047	27
F13	(	050	28
F14	)	051	29
F15	*	052	2A
F16	+	053	2B
F17	,	054	2C
F18	–	055	2D
F19	.	056	2E
F20	/	057	2F
F21	0	060	30
F22	1	061	31

Appendix C. Programming Considerations for Model 610 Tape Cassette System

C.1. GENERAL

The communications control procedures for the SPERRY UNIVAC Model 610 Tape Cassette System are in accordance with those for the UTS 400 terminal system. It is assumed that the programmer is familiar with the information contained in this manual, and particularly with the discussion of communications protocol given in Section 3. Refer to the current version of the tape cassette system concept and applications manual, UP-8282, for a functional description of tape cassette system operations.

The tape cassette system is a high-volume storage device that records data on either of two magnetic tape cassettes, which are mounted in two separate tape transports. The tape cassette system is connected to the UTS 400 through the 7-bit peripheral interface. Each tape transport requires two DIDs: one to select the write function, and one to select the read function. These DIDs may be any of the 12 valid DID characters in rows 3 through 14 of column 7 of the ASCII code chart in Figure 2–2.

The tape cassette system performs read, write, search, backward-one-block, and report-address operations as a result of commands from the UTS 400 operator or the host processor. These operations are performed through a combination of a device selection and a peripheral initiation command (4.6.1). Essentially, there are two modes of host-processor-controlled operation:

1. A selection poll is sent by the host processor, selection status is returned, and then a peripheral initiation command is sent by the host processor in a text message, or

2. The selection and peripheral initiation commands are combined in the form of a text message containing a specific DID and an initiation command.

C.2. STATUS REPORTING

When the selection reaches the tape cassette system, the device is deselected and the supplied DID is compared with the four cassette system DIDs. If there is a match, the tape cassette system is selected and its condition is reported to the peripheral interface by means of a 4-bit status code. The status codes sent to the interface and the corresponding UTS 400 responses to the host processor are listed in Table C–1.

Table C–1. Tape Cassette System Status Codes

Response Number	Mode	Status Code Bit Configuration (cassette system to peripheral interface)				Meaning	UTS 400 Response to Host 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	End of tape	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	End of tape	DLE <
7	Read	0	1	0	1	Data error	DLE :
8	Read	0	0	0	0	Device unable to proceed/no response	DLE =

In Table C–1, responses 1 and 5 indicate that the designated tape transport is ready to perform an operation.

Responses 2 and 6 indicate that the designated tape transport is at the end of tape.

Responses 3 and 7 indicate that the previous operation had a data error.

Responses 4 and 8 indicate a no-response condition. Possible reasons for this condition 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.
- There are problems associated with beginning-of-tape or end-of-tape procedures.
- A new selection was attempted while the cassette system was reading or writing a data block.
- 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.

C.3. HOST-CONTROLLED WRITE OPERATION

Data to be written is sent from the UTS 400 to the tape cassette system by means of a specific write DID and a peripheral initiation command (4.6.1). The host processor must then select the tape cassette system again to determine if the write operation occurred properly. The UTS 400 station reports a THRU condition to the host processor as soon as data transfer to the selected tape transport is completed. However, at this time the tape cassette system has not completed the block termination sequence, which requires approximately 160 milliseconds. If a specific selection of the same tape transport is attempted before the tape cassette system has completed the block termination sequence, a device-unable-to-proceed/no-response status (4 or 8, Table C–1) will be returned and the tape cassette system will be deselected.

If the UTS 400 automatic retry function is enabled, the THRU response indicates that the write operation was successfully completed. Therefore, a selection poll following the operation to verify its success is not required.

C.4. HOST-CONTROLLED READ OPERATION

The basic sequence used for write operations (C.3) is followed to read data from tape, except that a read selection (read DID) is specified. The text portion of the read selection contains only the peripheral initiation command. The THRU response indicates the read operation was completed successfully.

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

C.5. AUTOMATIC TRANSMIT CAPABILITY

The automatic transmit function is used where provisions for this function are included in the host processor program. The purpose of this function is to permit the host processor to read from the tape cassette system without having the host processor or UTS 400 operator initiate the transmit function for each block of data.

Although the tape cassette system has an automatic transmit function (enabled by the AUTO TR switch), this tape cassette system capability cannot be used by the UTS 400. Instead, the automatic transmit function is accomplished by means of the UTS 400 control page. Operator instructions for implementing the automatic transmit function in the control page are given in the operator's guide, UP-8358. Instructions for host-processor-initiated setting of the automatic transmit function in the control page are given in 4.5.12 of this manual.

C.6. HOST PROCESSOR CONTROL COMMANDS

Host processor control commands may be used to perform the following tape cassette system operations:

- Backward one block (control code BS)
- Report address (control code VT)
- Search/rewind (control code CAN)

The control command is located in the data portion of the host processor text message and is conveyed from the UTS 400 to the tape cassette system by means of a peripheral initiation command. If the first character received by the tape cassette system following a read selection is one of the control command codes (BS, VT, or CAN), the specific operation for that command is performed. (If no control command code is present, a normal read operation is performed.)

C.6.1. Backward-One-Block Command

A BS code following a read selection specifies the backward-one-block operation. Normally, the BS code is placed in the first character position after the SOE character located nearest and to the left of the cursor (or in the home position if no SOE is used) and is followed by a peripheral initiation command.

The UTS 400 reports a THRU condition in the poll response to indicate completion of the operation.

C.6.2. Report-Address Command

A VT control code following a read selection specifies the report-address operation. Normally, the VT code is placed in the first character position after the SOE character located nearest and to the left of the cursor (or in the home position if no SOE is used) and is followed by a peripheral initiation command. The tape cassette system then supplies the address to the UTS 400.

The UTS 400 reports a THRU condition in the poll response and then, if automatic transmit is set, supplies the address in a text message in the next traffic poll response.

C.6.3. Search Command

A CAN control code following a read selection specifies the search operation. Normally, the CAN code is placed in the first character position following the SOE character located nearest and to the left of the cursor (or in the home position if no SOE is used). The next characters following the CAN control code are interpreted by the tape cassette system as the mode, track, address, and data identifiers. The search command sequence is followed by a peripheral initiation command.

The tape cassette system performs the specified search and the UTS 400 reports a THRU condition in the poll response upon completion of the operation. If a data block was read as part of the search operation, that data is sent by the UTS 400 in the response to the next traffic poll if the control page has automatic transmit set or if the operator presses the XMIT key.

The search command sequence is as follows:

CAN mtaaaa/i . . . i

where:

CAN
: specifies a search operation

m
: designates the search mode (@, A, B, or C)

t
: designates the track (1 or 2)

a
: designates the address (four digits) for modes @, A, and B

/
: designates the beginning of the data identifiers

i
: designates the data identifiers (up to 16) for modes B and C only

Four search modes may be specified for **m**:

@

causes a high-speed search to the specified tape-position address. No automatic data read occurs with this search. (An @ type search to address 0000 is a rewind operation.)

A

causes a high-speed search that locates and reads a block. The command-supplied address is compared with that written in the data block heading on the tape. The block is then read automatically.

B

causes a high-speed search to the vicinity specified by the tape-position block whose first characters (up to 16) are the same as those supplied in the identifier portion of the command. The block is then read automatically.

C

causes a read-speed search from the current tape position for the identifiers supplied in the command, followed by a read of the block in question. No high-speed operation is involved. The address characters in the command sequence are ignored.

NOTE:

The tape cassette system recognizes the following additional characters as mode designators in the search command sequence:

0 *or* **\`** *in place of* **@**

1 *or* **a** *in place of* **A**

2 *or* **b** *in place of* **B**

3 *or* **c** *in place of* **C**

C.7. END-OF-TAPE PROCEDURES

A hole in the tape or clear leader is used to indicate end of tape on read operations, but not on write operations. In a write operation, the end-of-tape signal is generated at a tachometer count of 6000.

Two end-of-tape 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 end-of-tape condition (when a tachometer count of 6000 is reached), the following tape cassette system sequence occurs:

1. Normal completion of the write operation in process.

2. Retention of the status at completion of the last block.

If no data errors were encountered during the operation in which the end-of-tape condition was detected, subsequent write selections (by means of a write selection poll, TEXT/SD message, or TEXT/SD/PI message) or read selections (by means of a read selection poll or TEXT/SD message) of that transport will result in an end-of-tape status response (2 or 6, Table C–1) until the tape is repositioned. However, subsequent read selections by means of a TEXT/SD/PI message will result in a sustained busy condition until the tape is repositioned. When the tape is already at the end-of-tape position, a TEXT/PI message with a read selection will also result in a sustained busy condition. A repositioning command (either backward-one-block or search) may be contained in either a TEXT/PI message or a TEXT/SD/PI message.

If a data error was encountered during the operation in which the end-of-tape condition was detected, a subsequent selection of the transport following the sustained busy condition will result in a data-error response (3 or 7, Table C–1), which is cleared after being reported online. Subsequent selections will result in end-of-tape status or a sustained busy condition as described previously.

If the tape happens to stop with the end-of-tape hole directly over the end-of-tape sensor, the tape cassette system interprets the end-of-tape hole as clear leader and halts operation. An offline rewind is then required to continue operation.

C.8. ERROR CONDITIONS AND RECOVERY PROCEDURES

A special error code is provided in error detection. This code is usually the blinking start marker code (FS), but the user may specify a different code. The selected code is strapped in by the Sperry Univac customer engineer at the user site.

C.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 peripheral 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 deselects. If automatic retry has been specified, the UTS 400 attempts to reread the block. If the read is successful, a THRU condition is reported by way of a poll response. If the retries are unsuccessful, or if automatic retry was not enabled, the peripheral interface is left in a sustained busy condition. At the next selection for this station, the sustained busy condition is cleared and this tape cassette transport returns a data-error response (7, Table C–1) to the peripheral interface. 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.

The host processor can read the block again (perform its own retry or re-retry) by performing a backward-one-block or search, followed by a new read command.

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.

C.8.2. Write Errors

Two types of write errors are detected by the tape cassette system: timing errors and parity errors on tape (through the read after write option). 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 deselects. If an error occurs, either timing or parity, the UTS 400 attempts to rewrite the block if an automatic retry is enabled. If the rewrite is successful, a THRU condition will be reported by way of a poll response. If the retries are unsuccessful, or automatic retry is not enabled, the peripheral interface will be left in a sustained busy condition. At the next selection for this station, the sustained busy condition is cleared and the tape cassette system returns a data-error response (3, Table C–1) to the peripheral interface. After the data-error response has been returned, the error status in the tape 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. The host processor can also retry by performing a backward-one-block operation (or search to a backward position) and then attempting to rewrite the data block containing the error. Without automatic retry, a selection must be performed following each THRU response to verify that the write operation occurred without error.

C.8.3. Improper-Selection Errors

If a new selection of the tape cassette system is attempted while a block is being written or read on that same station, the system automatically deselects 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 C–1). 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 C–1). A new selection of the halted transport will result in the return of data-error status (3 or 7, Table C–1). After the data-error status is reported, the error status is cleared; with the next selection, device-ready status will be returned.

C.9. NOTES AND PRECAUTIONS

The following information should be kept in mind when the tape cassette system is being used with the UTS 400.

1. For operation with the UTS 400, the tape cassette system should be strapped for an erase display sequence (ESC e ESC M).

2. If a selection is followed by a text message with a specific 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 upon reaching read or write load point.

3. 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 (C.6.1) 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.

4. At the end of each write operation, a block termination sequence is placed on the tape. The UTS 400 station reports a THRU condition to the host processor as soon as data transfer to the tape cassette system is completed. However, at this time the tape cassette system has not completed the block termination sequence, which requires approximately 160 milliseconds. If a selection is attempted before the tape cassette system has completed the block termination sequence, an improper-selection error occurs. The result of such a selection attempt is described in C.8.3.

5. 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.

6. After a rewind, it is recommended that a report-address command be issued to position the tape at read load point.

7. A mode @ search to address 0000 will initiate a rewind. If automatic retry is enabled for host processor operation, the tape will be positioned at the read load point. If a write operation is to be performed, a dummy block should be written first to move the tape to the write load point, where the first block of data should begin.

8. The mode @ search may not stop the transport exactly on the address specified. A report-address operation should be performed before any further write operations.

9. When the mode @ search is used, the next read may be from the middle of a block and that read may create an error. To avoid this problem, follow the mode @ search with a backward-one-block operation to find a valid interblock gap, and note the address of this block for future mode A searches.

10. When tape cassettes are to be reused, degaussing is recommended to ensure that the information is entirely removed and the tapes are magnetically clean.

11. When a mode @ search is used to position a tape, a circuit in the tape cassette system is conditioned to adjust the length of the interblock gap for a write on the cassette following the search. If 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.

12. After the end-of-tape condition has been reported, the end-of-tape status may be cleared by initiating a device deselection DID (r) or pressing the STOP switch on the cassette system. The tape will move to clear leader on the next operation. When the tape is on clear leader, a rewind (pressing the tape cassette system REWIND switch) is required to continue operation.

    The end-of-tape status need not be cleared before performing certain valid operations such as backward-one-block, report-address, or search to an appropriate address. For example, a selection poll with a read DID that solicits end-of-tape status can be followed (after acknowledgment) with a TEXT/PI message containing the backward-one-block command.

13. The backward-one-block function, and a read operation when there is no data, cannot be interrupted by host processor control except by issuing a peripheral override DID (q) or a deselection DID (r). This process, however, master clears the tape cassette system and immediately halts operation.

14. The record separator function of the tape cassette system allows the ASCII record separator codes FS, GS, RS, and US to be written on tape within a block of data. Four data characters are specified to represent these codes and are decoded and translated to the four record separator codes and written on tape by the tape cassette system. The data characters specified are implemented by strapping in the tape cassette system.

    When the record separator codes are read back from the tape, record separator symbols appear on the UTS 400 screen.

The data characters specified for use with the UTS 400 and their corresponding record separator codes and symbols are as follows:

ASCII Character Entered on Screen	Record Separator Code Recorded on Tape	Symbol Produced by Record Separator Code Read from Tape
{	FS (file separator)	◤ (start blinking marker)
}	GS (group separator)	◥ (end blinking marker)
~	RS (record separator)	▶ (SOE character)
DEL (not displayed)	US* (unit separator)	FCC translation (not displayed)

Tapes prepared on UNISCOPE terminals using the record separator function may include US record separator codes that will alter the screen presentation from the original format when read into the UTS 400. These tapes may require editing to remove the US codes before the tapes can be used with the UTS 400.

15. The protected format function of the tape cassette system is not used with the UTS 400. The PROT FORMAT switch on the tape cassette system must be in the OFF position.

**The US code is reserved for FCC translation and should be strapped in the tape cassette system to be represented by a DEL character, which cannot be generated from the keyboard and should not be used by the host processor.*

Appendix D. Programming Considerations for Printers

D.1. GENERAL INFORMATION

The information in this appendix covers the SPERRY UNIVAC Communications Output Printer (COP), the SPERRY UNIVAC Model 800 Terminal Printer, the SPERRY UNIVAC 0786 Printer Subsystem, and the SPERRY UNIVAC 0791 Correspondence Quality Printer Subsystem. Individually, the printers are referred to as the COP, the terminal printer, the 0786 printer, and the 0791 printer; when references apply to all, the term "printer" or "printers" is used.

Since more than one printer can be connected to the peripheral interface, a specific device identifier (DID) code must be assigned to each printer. The DID may be any of the 12 valid DID characters in rows 3 through 14 of column 7 of the ASCII chart (Figure 2–2). The characters in rows 0, 1, and 15 are excluded.

The COP motor is started when the device is selected or when data is sent to the COP. After a selected time interval (which is strappable), if no data is being received by the COP, the COP motor is turned off. The 0786 printer motor is started when the device is selected or when a command function is sent to the printer. If no data is being received by the 0786 printer, the motor is turned off after 10 seconds.

D.2. COMMUNICATIONS OUTPUT PRINTER (COP)

D.2.1. COP Status Signal

Upon being selected by the host processor as the output peripheral, the COP reports its operating status to the peripheral interface by means of a status code. The status codes and their meanings are listed in Table D–1.

Table D–1. COP Status Codes

Description	Peripheral Interface to Processor (Poll Response)
Device ready	DLE >
Unable to proceed	DLE <
Not used	DLE :
No response	DLE =*

*"No response" is not actually sent by the COP but is produced by the peripheral interface as a result of no input data request from the COP.

The addressed COP will return the unable-to-proceed status if it is out of paper, if an incremental printer fuse is burned out, if the access cover is open, or if a print test operation is in progress. Otherwise, the device-ready status will be generated.

The UTS 400 performs a selection for every peripheral initiation. COPs strapped for line feed and carriage return on selection will perform a line feed and carriage return on each of these initiations.

D.2.2. Characters Printed by the COP

The ASCII chart (Figure 2–2) shows, in columns 2 through 5, the 64 characters that make up the character set on an ASCII print wheel and the 7-bit codes that cause each character to be printed. Because the print wheel of the incremental printer has only the 64 uppercase characters, the binary codes for the 32 lowercase characters in columns 6 and 7 will cause the character in the same row of column 4 or 5 to be printed. For example, if the code for b (column 6, row 2) is placed on the output data lines, a B (column 4, row 2) will be printed. Similarly, if the code for z (column 7, row 10) is presented, a Z (column 5, row 10) will be printed.

D.2.3. Control Characters Used by the COP

The COP recognizes three control characters from column 0 of the ASCII code chart (Figure 2–2). These are line feed (LF), paper advance or form feed (FF), and carriage return (CR). Upon receipt of the code for an LF or CR character, a 1-line paper advance occurs and the incremental printer carriage returns to the first character position (either character produces the same effect). An FF character also causes a carriage return, but the paper advances to the fourth line of the next form. Although the COP will accept continuous control characters, the paper feed solenoid fuse will burn out if paper is continuously advanced for approximately 19 seconds.

A carriage return operation can be initiated by placing one of the three control characters on the output data lines without regard to the print position of the carriage. In other words, the carriage can be returned from the first character position, from the 132nd character position, or from any position in between. A microswitch on the right tractor produces an automatic carriage return/line feed function if the carriage reaches the right tractor. This microswitch is intended as a safety device to prevent jamming of the mechanism in the absence of a carriage return or line feed command and is not intended for format control.

The carriage return operation is terminated when the left margin switch, a microswitch mounted on the left side of the incremental printer frame, is actuated by the carriage.

A field-strappable option is available that permits the COP to:

- Space on all control characters
- Not space on any control character
- Space on all control characters except one (user's choice)
- Not space on any control character except one (user's choice)

A field-strappable option is available that forces the COP to perform a carriage return and line feed operation at the end of a UTS 400 screen of data (user's choice).

Units are strapped for nonspace on any control character when shipped from the factory.

D.3. MODEL 800 TERMINAL PRINTER

D.3.1. Terminal Printer Status Signal

Upon being selected by the host processor as the output peripheral, the terminal printer reports its operating status to the peripheral interface by means of a status code. The status codes and their meanings are listed in Table D–2.

Table D–2. Terminal Printer Status Codes

Description	Peripheral Interface to Processor (Poll Response)
Device ready	DLE >
Unable to proceed	DLE <
Timing error	DLE :
No response	DLE =*

*"No response" is not actually sent by the terminal printer but is produced by the peripheral interface as a result of no input data request from the terminal printer.

The addressed terminal printer will return the unable-to-proceed status if it is out of paper; otherwise, the device-ready status will be generated.

The UTS 400 performs a selection for every peripheral initiation.

D.3.2. Characters Printed by the Terminal Printer

The ASCII chart (Figure 2–2) shows, in columns 2 through 7, the 96 characters that can be printed by the terminal printer, and the 7-bit codes corresponding to each character. Characters received from the terminal or the host processor are first entered into the 80-character buffer of the terminal printer. When the buffer is full or the applicable control character is received, the contents of the buffer are printed.

D.3.3. Control Characters Recognized by the Terminal Printer

The terminal printer recognizes three control characters from column 0 of the ASCII code chart (Figure 2–2). These are line feed (LF), paper advance or form feed (FF), and carriage return (CR). The terminal printer also recognizes an end-of-display sequence. When any of the three control characters or the end-of-display sequence is received, buffer loading ceases and the printer starts printing; after the data is printed, the printer carriage returns to the first character position and the paper advances one line. (The FF character causes a carriage return and a paper advance of 4 $\pm$1 lines.) The terminal printer can be strapped so that an end-of-display sequence received as the first character in a line may or may not cause the terminal printer to perform a line skip and advance the paper one line.

D.4. 0786 PRINTER SUBSYSTEM

D.4.1. 0786 Printer Status Signal

Upon being selected by the host processor as the output peripheral, the 0786 printer reports its operating status to the peripheral interface by means of a status code. The status codes and their meanings are listed in Table D–3.

Table D–3. 0786 Printer Status Codes

Description	Peripheral Interface to Processor (Poll Response)
Device ready	DLE >
Unable to proceed	DLE <
Timing error	DLE :
No response	DLE =*

*"No response" is not actually sent by the 0786 printer but is produced by the peripheral interface as a result of no input data request from the 0786 printer.

The addressed 0786 printer will return the unable-to-proceed status if it is out of paper; if it is in the process of printing a line, feeding paper, or clearing the line buffer; or in the case of a technical failure. Otherwise, the device-ready status will be generated.

The UTS 400 performs a selection for every peripheral initiation.

D.4.2. Characters Printed by the 0786 Printer

The 96 domestic (U.S.) characters that can be printed by the 0786 printer are shown in columns 2 through 7 of the ASCII chart (Figure 2–2) with the 7-bit codes corresponding to each character. Figure 2–4 shows the foreign-language characters this printer will produce. The characters are identified by the column and row in which they fit in the ASCII chart (Figure 2–2). The 0786 printer can also use a 128-character set consisting of the 64 Katakana characters shown in Figure 2–3 and the 64 uppercase characters shown in columns 2 through 5 of Figure 2–2. When the printer is using the English/Katakana character set, it must be connected to the UTS 400 by means of the 8-bit peripheral interface. Otherwise, it can be used with either the 7-bit or 8-bit interface.

Characters received from the terminal or the host processor are first entered into the line buffer of the 0786 printer. When the applicable control character is received, the contents of the buffer are printed.

D.4.3. Control Characters Recognized by the 0786 Printer

→ The 0786 printer recognizes the following eight control characters from columns 0 and 1 of the ASCII code chart (Figure 2-2):

→ Line feed (LF)

Vertical tab (VT)

Form feed (FF)

Carriage return (CR)

End of text (ETX)

Forward tab (HT)

Device control 3 (DC3)

Unit separator (US)

Printing of a line is initiated by a CR, LF, VT, ETX, or FF character. If the initiating control character is an LF or if it is a CR and the printer is strapped for the CR to cause a line feed, the printout is followed by a line space. If the initiating control character is a VT or an FF, a vertical-tab or a form-feed function is performed if the printer is equipped with a vertical format unit (VFU); if the printer has no VFU, the printout is followed by a line feed. If the printout is initiated by a CR character and the printer is not strapped for the CR to cause a line feed, no line advance is performed and the printhead remains in the last printed line position.

The 0786 printer is normally strapped for a CR character to cause a line feed; for a DC3, US, or HT character to cause a space; and for all other control characters to be ignored.

D.5. 0791 PRINTER SUBSYSTEM

D.5.1. 0791 Printer Status Signal

Upon being selected by the host processor as the output peripheral, the 0791 printer reports its operating status to the peripheral interface by means of a status code. The status codes and their meanings are listed in Table D–4.

The addressed 0791 printer will return the unable-to-proceed status if it is out of paper or ribbon, if it is in the process of printing a line or feeding paper, and in case of a technical failure. Otherwise, the device-ready status will be generated.

The UTS 400 performs a selection for every peripheral initiation.

Table D–4. 0791 Printer Status Codes

Description	Peripheral Interface to Processor (Poll Response)
Device ready	DLE >
Unable to proceed	DLE <
Timing, data, or printer parameter error	DLE :
No response	DLE =*

*"No response" is not actually sent by the 0791 printer but is produced by the peripheral interface as a result of no input data request from the 0791 printer.

D.5.2. Characters Printed by the 0791 Printer

The 96 domestic (US) characters that can be printed by the 0791 printer are shown in columns 2 through 7 of the ASCII chart (Figure 2-2) with the 7-bit codes corresponding to each character. Table D-5 shows the foreign language characters this printer will produce. Foreign language characters in Table D-5 are identified by their ASCII chart octal positions.

Characters received from the terminal or the host processor are first entered into the line buffer of the 0791 printer. When the applicable control character is received or when the buffer is full, the contents of the buffer are printed.

Table D–5. Foreign Language Characters Printed by the 0791 Printer

Octal	043	100	133	135	136	140	174	175	176
USA	#	@	[	]	∧	`	¦	}	∿
British*	£	@*	[	]	↑	`	¦	}	∿
French	£	à	°	§	∧	ç	ù	è	¨
German	£	§	Ä	Ü	∧	ö	ä	ü	ß
Swedish/Finnish	#	@	Ä	Å	∧	ö	ä	ā	∿
Danish/Norwegian	£	@	Æ	Å	∧	Ø	œ	ā	∿
Spanish	¢	§	¡	¿	∧	Ñ	ñ	£	∿

*British Pica 10 and Elite 12 produce the ‰ in octal position 100.

D.5.3. Control Characters Recognized by the 0791 Printer

The 0791 printer recognizes the following control characters from columns 0 and 1 of the ASCII chart (Figure 2-2):

Line feed (LF)

Vertical tab (VT)

Form feed (FF)

Carriage return (CR)

Printing is initiated by an LF, VT, FF, or CR character. If the initiating control character is an FF, a form feed function is performed. If printing is initiated by a CR character and the printer is not strapped for the CR to cause a line feed, no line advance is performed and the printhead remains in the last printed line position. The printer is normally strapped for a CR character to cause a line feed. The VT character causes a line feed.

The 0791 printer spaces upon receipt of an HT, DC3, DEL, or US control character and ignores all other control characters except LF, VT, FF, and CR.

The 0791 printer automatically prints the full buffer in response to a buffer overflow condition. Printing of the contents of the buffer is followed by a carriage return or by a line advance as determined by the printer internal strapping.

D.5.4. Format Control

The 0791 printer parameters can be altered with the following format control characters:

)ZCQP(FXXX:TXXX:CXX:LX:D1:END

where:

)ZCQP(

is the first word identifier that indicates printer parameter data. The first parenthesis (right) must be in the first character position of the print line.

END

is the second word identifier that indicates printer parameter data. It must be followed by a control character (LF, VT, FF, or CR).

F

designates the absolute form length in terms of six lines per inch. Even though the LPI switch may be set to 8, this parameter, when programmed, is interpreted as six lines per inch. For instance, a 14-inch form length would be specified as F84 (6 x 14 = 84). If the LPI switch were set to 8 LPI, a 14-inch form length parameter would still be designated as F84 (6 x 14 = 84) rather than F112 (8 x 14 = 112).

T

designates the text length in lines per inch. All parameter settings are scanned and altered (if a new entry or switch setting was made) following a top-of-form operation. The TXXX default parameter works in conjunction with the FXXX parameter as follows:

When FXXX = 66, then TXXX = 60 with a 1-inch perforation skip

When FXXX = 88, then TXXX = 80 with a 1-inch perforation skip

When the FXXX parameter is changed, the TXXX parameter will automatically change to allow a 1-inch perforation skip, regardless of the LPI switch setting (6 or 8) or the LX entry made.

However, the TXXX parameter may be changed to allow a larger or smaller perforation skip without affecting the FXXX parameter.

Once the TXXX parameter is changed from the default condition, it will not automatically default to 1 inch less than the form length parameter. A change in the FXXX parameter after the TXXX default condition is suspended requires that the TXXX parameter be changed.

C

designates characters per inch. Allowable values are 10 and 12.

L

designates the number of lines per inch. The allowable entries are 6 and 8. When this parameter is changed, the text length in inches may vary as follows:

If TXXX = 48 lines at 6 LPI, then total length = 8 inches

If TXXX = 48 lines at 8 LPI, then total length = 6 inches

D1

causes the printer to return to preset values (those strapped and set on the control panel). If the C and L parameters are changed, the LPI and CPI switches on the front panel are disabled until a D1 is received.

NOTE:

To determine the form length parameter (FXXX) entry for use with the cut sheet feeder, first determine the desired form length in inches and add 0.6 inch. Round the number to the next highest whole number and set the paper length switch on the cut sheet feeder to that number. Next, multiply the number of the paper length switch setting by 6 and enter the result into the FXXX parameter.

Example:

Form Length	Add 0.6 in.	Total	Rounded Up	Switch Setting	Multiply By 6	Parameter
11 in.	0.6 in.	11.6	12	12	72	F72
12.5 in.	0.6 in.	13.1	14	14	84	F84
14 in.	0.6 in.	14.6	15	15	90	F90

The FXXX value will not change unless printer power is turned off, or a new FXXX entry is made.

Entries may be made only in the parameters to be changed rather than change or reestablish all parameters.

Appendix E. Programming Considerations for Diskette Subsystem

E.1. GENERAL

The communications control procedures for the SPERRY UNIVAC Diskette Subsystem Type 8406 (diskette subsystem) are in accordance with those for the UTS 400. It is assumed that the programmer is familiar with the information contained in this manual, particularly with the discussion of communications protocol given in Section 3. Refer to the current version of the diskette subsystem general description, UP-8475, for a functional description of the device.

The diskette subsystem is a low-performance mass-storage device that records data on a magnetic diskette in a disk drive. Two disk drives may be included in one freestanding cabinet with one disk-drive controller. The device is connected to the UTS 400 through the 8-bit peripheral interface. Each disk drive requires two DIDs: one to select the write function, and one to select the read function. These DIDs may be any of the 12 valid DID characters in rows 3 through 14 of column 7 of the ASCII code chart in Figure 2–2.

The diskette subsystem performs read, write, search, backward-one-block, report-status, prepare-track, and report-address operations as a result of commands from the UTS 400 operator or the host processor. These operations are performed through a combination of a device selection and a peripheral initiation command (4.6.1). Essentially, there are two modes of host-processor-controlled operation:

1. A selection poll is sent by the host processor, selection status is returned, and then a peripheral initiation command is sent by the host processor in a text message, or

2. The selection and peripheral initiation commands are combined in the form of a text message containing a specific DID and an initiation command.

The diskette is addressed by track and sector. The accessible tracks are numbered 00 through 73. Each track has 26 sectors numbered 01 through 26. Thus, an address for track 42, sector 15, would be written as 4215. However, the UTS 400 also works with the tape cassette system, and the diskette and cassette system addressing methods must be compatible. Therefore, the total diskette address requires a dummy character before the address digits to position these digits in the same scheme as that required by the tape cassette system; for example, 14215.

An address for track 00, sector 01, is equivalent to home position. An address for track 00, sector 00, is converted to track 00, sector 01. In addition, any sector address greater than 26 causes the track number to be incremented by 1 and the sector number to be set to 01.

A block of data in the UTS 400 can be from 1 to 3744 characters long. The diskette subsystem writes each data block on sequential sectors, using as many sectors and tracks as necessary to accommodate the complete block. Only complete sectors are used; no sector is shared with any other block.

E.2. STATUS REPORTING

When the selection reaches the UTS 400, the supplied DID is compared with the disk drive DIDs. If there is a match, the device is selected and its condition is reported to the UTS 400 by means of an 8-bit status code. The status codes are interpreted by the UTS 400 and sent to the host processor. These status codes are listed in Table E–1.

Table E–1. Diskette Subsystem Status Codes Sent to Host Processor

Meaning of Code for Diskette Subsystem	UTS 400 Response to Host Processor
Device ready	DLE >
End of diskette or file security check	DLE <
Data error, cyclic redundancy check (CRC) error, or address error	DLE :
Device not ready (unable to proceed or no response)	DLE =

Possible reasons for the device-not-ready condition are:

- Diskette subsystem power is not on.
- The diskette subsystem is in an interlock condition because a diskette was not inserted properly or the disk drive door was not closed.
- A write function was commanded on a write-protected diskette.

E.3. HOST-CONTROLLED WRITE OPERATION

Data to be written is sent from the UTS 400 to the diskette subsystem by means of a specific write selection (write DID) and a peripheral initiation command (4.6.1). The UTS 400 station reports a THRU condition to the host processor as soon as data transfer to the selected disk drive is completed.

The first byte of each sector is used to control sector sequence within a block.

E.4. HOST-CONTROLLED READ OPERATIONS

The same basic sequence used for write operations (E.3) is followed to read data from disk, except that a read selection (read DID) is specified. The text portion of the read selection contains only the peripheral initiation command. The THRU response indicates that the read operation was completed successfully.

The first byte of each sector is stripped from the data block before the data is put into memory or on the screen.

E.5. AUTOMATIC TRANSMIT CAPABILITY

The automatic transmit function is used where provisions for this function are included in the host processor program. The purpose of this function is to permit the host processor to read from the diskette subsystem without having the host processor or UTS 400 operator initiate the transmit function for each block of data.

The automatic transmit function is accomplished by means of the UTS 400 control page. Instructions for operator use of the automatic transmit function are given in the UTS 400 operator's guide, UP-8358. Host-initiated setting of the automatic transmit function in the control page is described in 4.5.12.

E.6. HOST PROCESSOR CONTROL COMMANDS

Host processor control commands may be used to perform the following diskette subsystem operations:

- Backward one block (control code BS)
- Report address (control code VT)
- Search (control code CAN)

The read DID must be used in a text message containing a control command. The control command is located after the STX in the text message and is interpreted by the UTS 400 upon receipt of a peripheral initiation command (4.6.1). If the first character interpreted is one of the control command codes (BS, VT, or CAN), the specific operation for that command is performed. If no control command code is present, a normal read operation is performed. ←

E.6.1. Backward-One-Block Command

A BS code following a read selection specifies the backward-one-block operation. Normally, the BS code is placed in the first position following the SOE character located nearest and to the left of the cursor (or in the home position if no SOE is used) and is followed by a peripheral initiation command.

The UTS 400 reports a THRU condition in the poll response to indicate completion of the operation.

E.6.2. Report-Address Command

A VT control code following a read selection specifies the report-address operation. Normally, the VT code is placed in the first position following the SOE character located nearest and to the left of the cursor (or in the home position if no SOE is used) and is followed by a peripheral initiation command. The UTS 400 will place the current diskette address on the screen at the cursor location.

The UTS 400 reports a THRU condition in the poll response to indicate completion of the operation.

E.6.3. Search Command

A CAN control code following a read selection specifies the search operation. Normally, the CAN code is placed in the first position following the SOE character located nearest and to the left of the cursor (or in the home position if no SOE is used). The next characters following the CAN control code are interpreted as the mode, address, and data identifiers. The search command sequence is followed by a peripheral initiation command.

The diskette subsystem performs the specified search and the UTS 400 reports a THRU condition in the poll response upon completion of the operation. If a data block was read as part of the search operation, that data is sent by the UTS 400 in the response to the next traffic poll if the control page has automatic transmit set or if the operator presses the XMIT key.

The search command sequence is as follows:

CAN mtaaa/i . . . i

where:

CAN
specifies a search operation

m
designates the search mode (@, A, B, or C)

t
designates a dummy character used to hold the position of the digits following

aaaa
designates the address (four digits) for search modes @, A, and B, with the first two digits specifying the track (00 through 73) and the last two digits specifying the sector of the disk (01 through 26)

/
designates the beginning of the data identifiers (for search modes B and C only)

i . . . i
designates the data identifiers (up to 16) for modes B and C only

Four search modes may be specified for **m**:

@
specifies a search to the track and sector indicated in the address. No automatic data read occurs with this mode. (An @ type search to address 10000 or 10001 is a home operation.)

A
specifies a search to the indicated track and sector followed by a read of a block.

B
specifies a search to the command-supplied address followed by a comparison of the first data characters in the successive blocks against those supplied in the identifier portion of the command. When the match is made, the block is read automatically.

C

specifies a search from the current disk position for the identifiers supplied in the command, followed by a read of the block when it is located. Any address digits in the command sequence are ignored, but one dummy character must be placed in the address area.

NOTE:

The diskette subsystem recognizes the following additional characters as mode designators in the search command sequence:

0 *or* \` *in place of* @

1 *or* a *in place of* A

2 *or* b *in place of* B

3 *or* c *in place of* C

E.7. END-OF-DISK PROCEDURES

A particular track and sector number indicates end-of-disk on read and write operations. For a write, the end-of-disk condition is identified when an address of 17222 is detected, and for a read, when an address of 17400 is detected.

When the UTS 400 detects the end-of-disk condition, the following sequence occurs:

1. Normal completion of the write or read operation in process

2. Retention of the status at completion of the last block

The end-of-disk status cannot and need not be cleared prior to performing any valid operation (backward one block, report address, search to an acceptable address). For example, a selection poll with a read DID that receives end-of-disk status can be followed (after acknowledgment) with a backward-one-block command with a general DID.

If no data errors were encountered during the operation in which the end-of-disk condition was detected, subsequent write selections (by means of a write selection poll, a TEXT/SD message, or a TEXT/SD/PI message) or read selections (by means of a read selection poll or a TEXT/SD message) of that disk drive will result in an end-of-disk status response until the read/write head is repositioned. However, the first read selection by means of a TEXT/SD/PI message will result in a sustained busy condition, and subsequent selections will result in end-of-disk status until the read/write head is repositioned.

If a data error was encountered during the operation in which the end-of-disk condition was detected, a subsequent selection of the disk drive will result in a data-error response, which is cleared after being reported online. Subsequent selections will result in end-of-disk status or a sustained busy condition, as described in the preceding paragraph.

E.8. ERROR CONDITIONS AND RECOVERY PROCEDURES

E.8.1. Read Errors

If a data error (either a parity or timing error) occurs during a read operation, the read is discontinued and the diskette subsystem will be automatically deselected. If automatic retry has been specified, the UTS 400 attempts to reread the block. If the read is successful, a THRU condition will be reported by way of a poll response. If the retries are unsuccessful, or if automatic retry was not enabled, the peripheral interface will be left in a sustained busy condition. At the next selection for this station, the sustained busy condition will be cleared and the current status of the diskette subsystem returned.

NOTE:

No selection is ever required following the THRU response, even though automatic retry is not enabled.

The host processor can read the block again (perform its own retry or re-retry) by performing a backward-one-block or search operation, followed by a new read command.

If the diskette subsystem repeatedly returns an error response at the same location on the disk, the disk is probably damaged or dirty at that point.

E.8.2. Write Errors

Two types of write errors are detected by the diskette subsystem: timing errors and parity errors on disk (through the read-after-write function). If a timing error occurs during a write operation, the error status is retained and the diskette subsystem will be automatically deselected. If either a timing or parity error occurs during a write operation, the UTS 400 will attempt to rewrite the block if automatic retry is enabled. If the rewrite is successful, a THRU condition will be reported by way of a poll response. If the retries are unsuccessful, or if automatic retry is not enabled, the UTS 400 will be left in a sustained busy condition. At the next selection for this station, the sustained busy condition will be cleared and the current status of the diskette subsystem returned. With the next selection, the data block can be rewritten or the normal write operation can be continued.

NOTE:

No selection is ever required following the THRU response, even though automatic retry is not enabled.

The host processor can also retry by performing a backward-one-block operation (or search to a backward position) and then attempting to rewrite the data block containing the error.

CAUTION

If the write attempt had not started, the backward-one-block operation may position the read/write head to the beginning of a previous block.

E.8.3. Improper-Selection Errors

If a new selection of the diskette subsystem is attempted while a block is being written or read on that same station, the subsystem will be automatically deselected. When the interruption occurs during a write operation, only the character being written is completed. When the interruption occurs during a read operation, data transfer stops. If the interruption is a selection poll, the device-ready status (DLE >) will be returned.

E.9. NOTES AND PRECAUTIONS

The following information should be kept in mind when the diskette subsystem is being used with the UTS 400.

1. A mode @ search to address 10000 or 10001 will return the disk drive to the home position.

2. When the mode @ search is used, the next read could occur in the middle of a block or the next write could begin over the end of a block. To avoid this problem, follow the mode @ search with a backward-one-block operation to find a start-of-block address.

3. If the end of a block has been overwritten, a read of that block will result in a sustained busy condition because the block lacks an end-of-block indication. A selection will be required to clear the sustained busy condition, after which the device-ready status (DLE >) will be returned.

4. Any DID except the general DID, if sent while an operation is in progress, will cause the diskette subsystem to be deselected.

5. The record separator function of the diskette subsystem allows the ASCII record separator codes FS, GS, and RS to be written on disk within a block of data. Three data characters are specified to represent the record separator codes. These three data characters are decoded and translated to the three record separator codes and written on disk by the diskette subsystem.

   When the record separator codes are read back from the disk, record separator symbols appear on the UTS 400 screen.

   The data characters used with the UTS 400 and their corresponding record separator codes and symbols are as follows:

ASCII Character Entered on Screen	Record Separator Code Recorded on Disk	Symbol Produced by Record Separator Code Read from Disk
}	FS (file separator)	◤(start blinking marker)
{	GS (group separator)	◥(end blinking marker)
~	RS (record separator)	▶ (SOE character)

Appendix F. Processor Instruction Set

F.1. GENERAL

The processor instruction set includes five types of instructions, as follows:

- Data transfer group – Move data between registers or between memory and registers.
- Arithmetic group – Add, subtract, increment, or decrement data in registers or in memory.
- Logical group – Perform AND, OR, exclusive OR, compare, rotate, or complement data operations in registers or in memory.
- Branch group – Perform conditional and unconditional jump instructions, subroutine call instructions, and return instructions.
- Stack, I/O, and machine control group – Perform I/O instructions and instructions for maintaining the stack and internal control flags.

F.2. INSTRUCTION AND DATA FORMATS

The processor memory is organized into quantities defined as bytes. Each byte has a unique 16-bit binary address corresponding to its sequential position in memory. The processor can address up to 65,536 bytes of memory, which consists of both read-only memory (ROM) elements and random-access memory (RAM) elements.

Data in the processor is stored in the form of 8-bit binary integers, with bit 0 designated as the least significant bit and bit 7 the most significant bit.

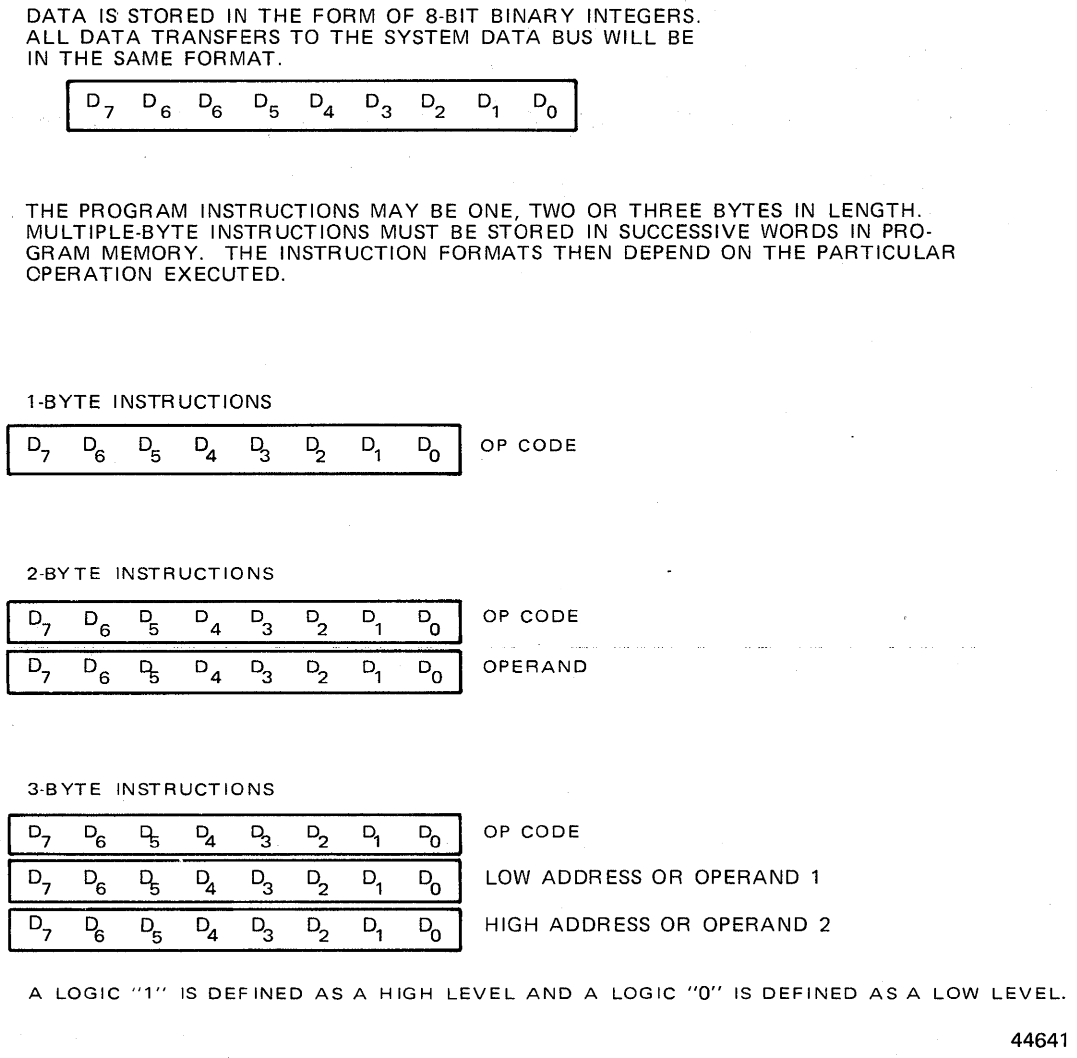
The processor program instructions may be one, two, or three bytes in length. Multiple-byte instructions must be stored in successive memory locations; the address of the first byte is always used as the address of the instruction. The instruction format depends on the instruction to be executed. Figure F–1 illustrates and defines the processor data and instruction formats.

F.3. ADDRESSING MODES

The processor has four modes for addressing data stored in memory or in registers, as follows:

- Direct (three bytes) – The memory address of the data item is contained in bytes 2 and 3 of the instruction. The low-order bits of the address are in byte 2 and the high-order bits in byte 3.
- Register (one byte) – The register or register pair containing the data specified in the instruction.

Portions of the material in this appendix are reprinted by permisssion of Intel Corporation.


DATA IS STORED IN THE FORM OF 8-BIT BINARY INTEGERS. ALL DATA TRANSFERS TO THE SYSTEM DATA BUS WILL BE IN THE SAME FORMAT.

THE PROGRAM INSTRUCTIONS MAY BE ONE, TWO OR THREE BYTES IN LENGTH. MULTIPLE-BYTE INSTRUCTIONS MUST BE STORED IN SUCCESSIVE WORDS IN PROGRAM MEMORY. THE INSTRUCTION FORMATS THEN DEPEND ON THE PARTICULAR OPERATION EXECUTED.

A LOGIC "1" IS DEFINED AS A HIGH LEVEL AND A LOGIC "0" IS DEFINED AS A LOW LEVEL.

D_7 D_6 D_6 D_5 D_4 D_3 D_2 D_1 D_0
1-BYTE INSTRUCTIONS
D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0
OP CODE
2-BYTE INSTRUCTIONS
D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0
OP CODE
D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0
OPERAND
3-BYTE INSTRUCTIONS
D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0
OP CODE
D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0
LOW ADDRESS OR OPERAND 1
D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0
HIGH ADDRESS OR OPERAND 2
44641

Figure F–1. Processor Data and Instruction Formats

- Register indirect (one byte) – A register pair containing the memory address where the data is located is specified in the instruction. The high-order bits of the address are in the first register of the pair and the low-order bits in the second.
- Immediate (two bytes) – The data itself is contained in the instruction. The data is either an 8-bit quantity or a 16-bit quantity. The least significant byte is first and the most significant byte second.

Unless otherwise directed by an interrupt or branch instruction, the execution of instructions proceeds through consecutively increasing memory locations. A branch instruction can specify the address of the next instruction to be executed in one of two ways:

- Direct – The branch instruction contains the address of the next instruction to be executed. Byte 2 contains the low-order address and byte 3 the high-order address. (The RST instruction is an exception.)
- Register indirect – The branch instruction indicates a register pair that contains the address of the next instruction to be executed. The high-order bits of the address are in the first register of the pair and the low-order bits in the second.

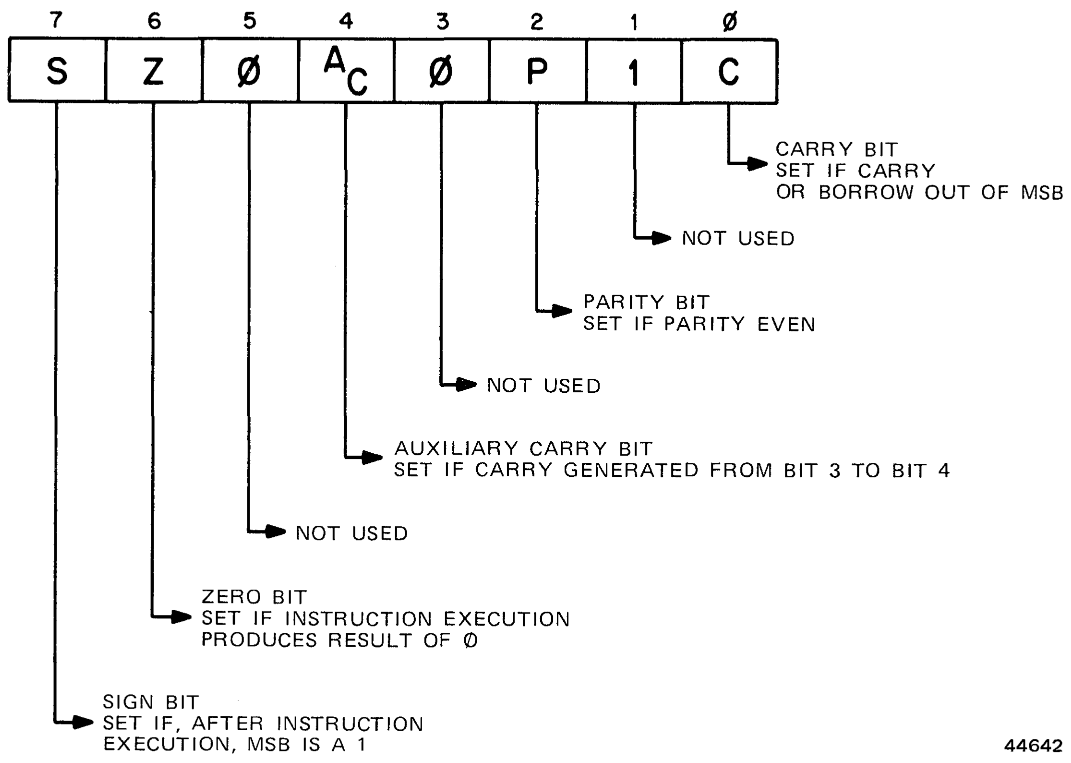
F.4. CONDITION FLAGS

Five condition flags are associated with the execution of instructions: zero, sign, parity, carry, and auxiliary carry. Each condition flag is represented by a 1-bit register in the processor. A flag is set by setting the bit to 1 and is reset by setting the bit to 0.

Unless otherwise indicated, when a flag is affected by an instruction, it is affected in the following manner:

- Zero (Z) – If the result of an instruction is a value of 0, this flag is set; otherwise, it is reset.
- Sign (S) – If the most significant bit of the result of the operation has a value of 1, this flag is set; otherwise, it is reset.
- Parity (P) – If the modulo 2 sum of the bits of the result of the operation is 0 (that is, if the result has even parity), this flag is set; otherwise, it is reset (that is, if the result has odd parity).
- Carry (CY) – If the instruction resulted in a carry (from addition) or a borrow (from subtraction or a comparison) out of the high-order bit, this flag is set; otherwise, it is reset.
- Auxiliary carry (AC) – If the instruction caused a carry out of bit 3 and into bit 4 of the resulting value, the auxiliary carry flag is set; otherwise, it is reset. This flag is affected by single-precision additions, subtractions, increments, decrements, comparisons, and logical operations, but is principally used with additions and with increments preceding a DAA (decimal adjust accumulator) instruction (as described in Table F–2).

Figure F–2 illustrates the flag byte stack format.



44642

Figure F–2. Flag Byte Stack Format

F.5. INSTRUCTION SET DESCRIPTION

Detailed descriptions for the five instruction set groups are provided in F.5.1 through F.5.5. Summary lists of the instruction set in alphabetical and hexadecimal sequence are supplied in F.5.6. The symbols and abbreviations used in the instruction set lists are illustrated in Figure F–3.

Use of certain instructions can create program conflicts when user programs are being prepared for the UTS 400. Restrictions applying to use of the instruction set under these conditions are defined in Section 5, paragraph 5.10.

F.5.1. Data Transfer Group Instructions

This instruction group is used to transfer data to and from registers and memory. Condition flags are not affected by any instruction in the data transfer group. Table F–1 contains a list and descriptions of the instructions.

F.5.2. Arithmetic Group Instructions

This instruction group is used to perform arithmetic operations on data in registers and memory. Unless otherwise indicated, all instructions in the arithmetic group affect the Z, S, P, CY, and AC condition flags. Table F–2 contains a list and descriptions of the instructions.

F.5.3. Logical Group Instructions

This instruction group is used to perform logical operations on data in registers and memory and on condition flags. Unless otherwise indicated, all instructions in the logical group affect the Z, S, P, CY, and AC condition flags. Table F–3 contains a list and descriptions of the instructions.

F.5.4. Branch Group Instructions

This instruction group is used to alter the normal sequential program flow. Condition flags are not affected by any instruction in the branch group.

The two types of branch instructions are unconditional and conditional. Unconditional transfers simply perform the specified operation on the program counter (PC) register. Conditional transfers examine the status of one of the four processor flags to determine if the specified branch is to be executed. The conditions that may be specified are:

Condition	Condition Code
NZ – not zero (Z = 0)	000
Z – zero (Z = 1)	001
NC – no carry (CY = 0)	010
C – carry (CY = 1)	011
PO – parity odd (P = 0)	100
PE – parity even (P = 1)	101
P – plus (S = 0)	110
M – minus (S = 1)	111

Table F–4 contains a list and descriptions of the instructions.

F.5.5. Stack, I/O, and Machine Control Group Instructions

This instruction group is used to perform input and output operations, manipulate the stack, and alter internal control flags. Unless otherwise specified, condition flags are not affected by any instruction in this group. Table F–5 contains a list and descriptions of the instructions.

F.5.6. Instruction Summaries

Summaries of the processor instruction set in alphabetical sequence and in hexadecimal code sequence are provided in Tables F–6 and F–7, respectively.

SYMBOL OR ABBREVIATION	MEANING
A	ACCUMULATOR
A_n	BIT n OF THE A-REGISTER
⟨ B2 ⟩	SECOND BYTE OF THE INSTRUCTION
⟨ B3 ⟩	THIRD BYTE OF THE INSTRUCTION
C	ONE OF THE FOLLOWING FLAG FLIP-FLOP REFERENCES:

FLAG FLIP-FLOPS	CONDITION FOR TRUE
CARRY (CY)	– OVERFLOW, UNDERFLOW
PARITY (P)	– PARITY OF RESULT IS EVEN
SIGN (S)	– MOST SIGNIFICANT BIT OF RESULT IS 1
ZERO (Z)	– RESULT IS ZERO

DDD DESTINATION REGISTER FOR DATA:

REGISTER NO. (SSS OR DDD)	REGISTER NAME	REGISTER NO. (SSS OR DDD)	REGISTER NAME
000	B	100	H
001	C	101	L
010	D	110	MEMORY
011	E	111	ACCUMULATOR

SYMBOL OR ABBREVIATION	MEANING
F	FLAG BYTE STACK
M	MEMORY LOCATION INDICATED BY CONTENTS OF REGISTERS H AND L
NNN	BINARY REPRESENTATION 000 THRU 111 FOR RESTART NUMBERS 0 THRU 7, RESPECTIVELY
PC	PROGRAM COUNTER
PCh	HIGH-ORDER BITS OF PC
PCl	LOW-ORDER BITS OF PC
PSW	PROCESSOR STATUS WORD
r, r1, r2	ONE OF THE SCRATCH-PAD REGISTER REFERENCES: A, B, C, D, E, H, L
rh	HIGH-ORDER REGISTER OF REGISTER PAIR rp
rl	LOW-ORDER REGISTER OF REGISTER PAIR rp
SP	STACK POINTER
SSS	SOURCE REGISTER FOR DATA (SEE MEANING FOR DDD ABOVE)
XXX	A "DON'T CARE"
( )	CONTENTS OF LOCATION OR REGISTER
$\wedge$	LOGICAL PRODUCT (AND)
$\vee$	LOGICAL SUM (INCLUSIVE OR)
$\veebar$	LOGICAL DIFFERENCE (EXCLUSIVE OR)
$\leftarrow$	IS TRANSFERRED TO
$\leftrightarrow$	IS EXCHANGED WITH
:	COMPARE

44643

Figure F–3. Symbols and Abbreviations

Table F–1. Data Transfer Group Instructions (Part 1 of 2)

Name	Mnemonic	Operation	Format	Cycles	States	Addressing
			D_7 D_0			
Move Register	MOV r1, r2	(r1) ← (r2) The content of register r2 is moved to register r1.	0 1 D D D S S S	1	5	Register
Move from Memory	MOV r, M	(r) ← ((H) (L)) The content of the memory location whose address is in registers H and L is moved to register r.	0 1 D D D 1 1 0	2	7	Register indirect
Move to Memory	MOV M, r	((H) (L)) ← (r) The content of register r is moved to the memory location whose address is in registers H and L.	0 1 1 1 0 S S S	2	7	Register indirect
Move Immediate	MVI r, data	(r) ← (byte 2) The content of byte 2 of the instruction is moved to register r.	0 0 D D D 1 1 0 Data	2	7	Immediate
Move to Memory Immediate	MVI M, data	((H) (L)) ← (byte 2) The content of byte 2 of the instruction is moved to the memory location whose address is in registers H and L.	0 0 1 1 0 1 1 0 Data	1	10	Immediate/ register indirect
Load Register Pair Immediate	LXI rp, data 16	(rh) ← (byte 3) (rl) ← (byte 2) Byte 3 of the instruction is moved into the high-order register (rh) of the register pair rp. Byte 2 of the instruction is moved into the low-order register (rl) of the register pair rp.	0 0 R P 0 0 0 1 Low-order data High-order data	3	10	Immediate
Load Accumulator Direct	LDA addr	(A) ← ((byte 3) (byte 2)) The content of the memory location whose address is specified in byte 2 and byte 3 of the instruction is moved to register A.	0 0 1 1 1 0 1 0 Low-order address High-order address	4	13	Direct
Store Accumulator Direct	STA addr	((byte 3) (byte 2)) ← (A) The content of the accumulator is moved to the memory location whose address is specified in byte 2 and byte 3 of the instruction.	0 0 1 1 0 0 1 0 Low-order address High-order address	4	13	Direct

Table F–1. Data Transfer Group Instructions (Part 2 of 2)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing
Load H and L Direct	LHLD addr	(L) ← ((byte 3) (byte 2)) (H) ← ((byte 3) (byte 2) + 1) The content of the memory location whose address is specified in byte 2 and byte 3 of the instruction is moved to register L. The content of the memory location at the succeeding address is moved to register H.	0 0 1 0 1 0 1 0 Low-order address High-order address	5	16	Direct
Store H and L Direct	SHLD addr	((byte 3) (byte 2)) ← (L) ((byte 3) (byte 2) + 1) ← (H) The content of register L is moved to the memory location whose address is specified in byte 2 and byte 3. The content of register H is moved to the succeeding memory location.	0 0 1 0 0 0 1 0 Low-order address High-order address	5	16	Direct
Load Accumulator Indirect	LDAX rp	(A) ← ((rp)) The content of the memory location whose address is in the register pair rp is moved to register A. NOTE: Only register pairs rp = B (registers B and C) or rp = D (registers D and E) may be specified.	0 0 R P 1 0 1 0	2	7	Register indirect
Store Accumulator Indirect	STAX rp	((rp)) ← (A) The content of register A is moved to the memory location whose address is in the register pair rp. NOTE: Only register pairs rp = B (registers B and C) or rp = D (registers D and E) may be specified.	0 0 R P 0 0 1 0	2	7	Register indirect
Exchange H and L with D and E	XCHG	(H) ↔ (D) (L) ↔ (E) The contents of registers H and L are exchanged with the contents of registers D and E.	1 1 1 0 1 0 1 1	1	4	Register

Table F–2. Arithmetic Group Instructions (Part 1 of 5)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
Add Register	ADD r	(A) ← (A) + (r) The content of register r is added to the content of the accumulator. The result is placed in the accumulator.	1 0 0 0 0 S S S	1	4	Register	Z, S, P, CY, AC
Add Memory	ADD M	(A) ← (A) + ((H) (L)) The content of the memory location whose address is contained in the H and L registers is added to the content of the accumulator. The result is placed in the accumulator.	1 1 0 0 0 1 1 0	2	7	Register indirect	Z, S, P, CY, AC
Add Immediate	ADI data	(A) ← (A) + (byte 2) The content of the second byte of the instruction is added to the content of the accumulator. The result is placed in the accumulator.	1 1 0 0 0 1 1 0 Data	2	7	Immediate	Z, S, P, CY, AC
Add Register with Carry	ADC r	(A) ← (A) + (r) + (CY) The content of register r and the content of the carry bit are added to the content of the accumulator. The result is placed in the accumulator.	1 0 0 0 1 S S S	1	4	Register	Z, S, P, CY, AC
Add Memory with Carry	ADC M	(A) ← (A) + ((H) (L)) + (CY) The content of the memory location whose address is contained in the H and L registers and the content of the CY flag are added to the accumulator. The result is placed in the accumulator.	1 0 0 0 1 1 1 0	2	7	Register indirect	Z, S, P, CY, AC

Table F–2. Arithmetic Group Instructions (Part 2 of 5)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
Add Immediate with Carry	ACI data	(A) ← (A) + (byte 2) + (CY) The content of the second byte of the instruction and the content of the CY flag are added to the content of the accumulator. The result is placed in the accumulator.	1 1 0 0 1 1 1 0 Data	2	7	Immediate	Z, S, P, CY, AC
Subtract Register	SUB r	(A) ← (A) – (r) The content of register r is subtracted from the content of the accumulator. The result is placed in the accumulator.	1 0 0 1 0 S S S	1	4	Register	Z, S, P, CY, AC
Subtract Memory	SUB M	(A) ← (A) – ((H) (L)) The content of the memory location whose address is contained in the H and L registers is subtracted from the content of the accumulator. The result is placed in the accumulator.	1 0 0 1 0 1 1 0	2	7	Register indirect	Z, S, P, CY, AC
Subtract Immediate	SUI data	(A) ← (A) – (byte 2) The content of the second byte of the instruction is subtracted from the content of the accumulator. The result is placed in the accumulator.	1 1 0 1 0 1 1 0 Data	2	7	Immediate	Z, S, P, CY, AC
Subtract Register with Borrow	SBB r	(A) ← (A) – (r) – (CY) The content of register r and the content of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.	1 0 0 1 1 S S S	1	4	Register	Z, S, P, CY, AC

Table F–2. Arithmetic Group Instructions (Part 3 of 5)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
Subtract Memory with Borrow	SBB M	(A) ← (A) – ((H) (L)) – (CY) The content of the memory location whose address is contained in the H and L registers and the content of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.	1 0 0 1 1 1 1 0	2	7	Register indirect	Z, S, P, CY, AC
Subtract Immediate with Borrow	SBI data	(A) ← (A) – (byte 2) – (CY) The content of the second byte of the instruction and the content of the CY flag are both subtracted from the accumulator. The result is placed in the accumulator.	1 1 0 1 1 1 1 0 Data	2	7	Immediate	Z, S, P, CY, AC
Increment Register	INR r	(r) ← (r) + 1 The content of register r is incremented by 1. NOTE: All condition flags except CY are affected.	0 0 D D D 1 0 0	1	5	Register	Z, S, P, AC
Increment Memory	INR M	((H) (L)) ← ((H) (L)) + 1 The content of the memory location whose address is contained in the H and L registers is incremented by 1. NOTE: All condition flags except CY are affected.	0 0 1 1 0 1 0 0	3	10	Register indirect	Z, S, P, AC

Table F–2. Arithmetic Group Instructions (Part 4 of 5)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
Decrement Register	DCR r	(r) ← (r) – 1 The content of register r is decremented by 1. NOTE: All condition flags except CY are affected.	0 0 D D D 1 0 1	1	5	Register	Z, S, P, AC
Decrement Memory	DCR M	((H) (L)) ← ((H) (L)) – 1 The content of the memory location whose address is contained in the H and L registers is decremented by 1. NOTE: All condition flags except CY are affected.	0 0 1 1 0 1 0 1	3	10	Register indirect	Z, S, P, AC
Increment Register Pair	INX rp	(rh) (rl) ← (rh) (r) + 1 The content of register pair rp is incremented by 1. NOTE: No condition flags are affected.	0 0 R P 0 0 1 1	1	5	Register	None
Decrement Register Pair	DCX rp	(rh) (rl) ← (rh) (rl) – 1 The content of register pair rp is decremented by 1. NOTE: No condition flags are affected.	0 0 R P 1 0 1 1	1	5	Register	None

Table F–2. Arithmetic Group Instructions (Part 5 of 5)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
Add Register Pair to H and L	DAD rp	(H) (L) ← (H) (L) + (rh) (rl) The content of register pair rp is added to the content of register pair H and L. The result is placed in register pair H and L. NOTE: Only the CY flag is affected. It is set if there is a carry out of the double precision add; otherwise, it is reset.	0 0 R P 1 0 0 1	3	10	Register	CY
Decimal Adjust Accumulator	DAA	The 8-bit number in the accumulator is adjusted to form two 4-bit binary-coded-decimal digits by the following processes: 1. If the value of the least significant four bits of the accumulator is greater than 9 or if the AC flag is set, 6 is added to the accumulator. 2. If the value of the most significant four bits of the accumulator is now greater than 9, or if the CY flag is set, 6 is added to the most significant four bits of the accumulator. NOTE: All flags are affected.	0 0 1 0 0 1 1 1	1	4		Z, S, P, CY, AC

Table F–3. Logical Group Instructions (Part 1 of 5)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
AND Register	ANA r	(A) ← (A) $\wedge$ (r) An AND operation is performed on the content of register r and the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.	1 0 1 0 0 S S S	1	4	Register	Z, S, P, CY, AC
AND Memory	ANA M	(A) ← (A) $\wedge$ ((H) (L)) An AND operation is performed on the content of the memory location whose address is contained in the H and L registers and the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.	1 0 1 0 0 1 1 0	2	7	Register indirect	Z, S, P, CY, AC
AND Immediate	ANI data	(A) ← (A) $\wedge$ (byte 2) An AND operation is performed on the content of the second byte of the instruction and the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.	1 1 1 0 0 1 1 0 Data	2	7	Immediate	Z, S, P, CY, AC
Exclusive OR Register	XRA r	(A) ← (A) $\veebar$ (r) An exclusive OR operation is performed on the content of register r and the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.	1 0 1 0 1 S S S	1	4	Register	Z, S, P, CY, AC

Table F–3. Logical Group Instructions (Part 2 of 5)

Name	Mnemonic	Operation	Format	Cycles	States	Addressing	Flags
			D_7 D_0				
Exclusive OR Memory	XRA M	(A) ← (A) $\veebar$ ((H) (L)) An exclusive OR operation is performed on the content of the memory location whose address is contained in the H and L registers and the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.	1 0 1 0 1 1 1 0	2	7	Register indirect	Z, S, P, CY, AC
Exclusive OR Immediate	XRI data	(A) ← (A) $\veebar$ (byte 2) An exclusive OR operation is performed on the content of the second byte of the instruction and the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.	1 1 1 0 1 1 1 0 Data	2	7	Immediate	Z, S, P, CY, AC
OR Register	ORA r	(A) ← (A) V (r) An inclusive OR operation is performed on the content of register r and the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.	1 0 1 1 0 S S S	1	4	Register	Z, S, P, CY, AC
OR Memory	ORA M	(A) ← (A) V ((H) (L)) An inclusive OR operation is performed on the content of the memory location whose address is contained in the H and L registers and the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.	1 0 1 1 0 1 1 0	2	7	Register indirect	Z, S, P, CY, AC

Table F–3. Logical Group Instructions (Part 3 of 5)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
OR Immediate	ORI data	(A) ← (A) ∨ (byte 2) An inclusive OR operation is performed on the content of the second byte of the instruction and the content of the accumulator. The result is placed in the accumulator. The CY and AC flags are cleared.	1 1 1 1 0 1 1 0 Data	2	7	Immediate	Z, S, P, CY, AC
Compare Register	CMP r	(A) – (r) The content of register r is subtracted from the accumulator. The accumulator remains unchanged. The condition flags are set as a result of the subtraction. The Z flag is set to 1 if (A) = (r). The CY flag is set to 1 if (A) < (r).	1 0 1 1 1 S S S	1	4	Register	Z, S, P, CY, AC
Compare Memory	CMP M	(A) – ((H) (L)) The content of the memory location whose address is contained in the H and L registers is subtracted from the accumulator. The accumulator remains unchanged. The condition flags are set as a result of the subtraction. The Z flag is set to 1 if (A) = ((H) (L)). The CY flag is set to 1 if (A) < ((H) (L)).	1 0 1 1 1 1 1 0	2	7	Register indirect	Z, S, P, CY, AC
Compare Immediate	CPI data	(A) – (byte 2) The content of the second byte of the instruction is subtracted from the accumulator. The condition flags are set by the result of the subtraction. The Z flag is set to 1 if (A) = (byte 2). The CY flag is set to 1 if (A) < (byte 2).	1 1 1 1 1 1 1 0 Data	2	7	Immediate	Z, S, P, CY, AC

Table F–3. Logical Group Instructions (Part 4 of 5)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
Rotate Left	RLC	$(A_{n+1}) \leftarrow (A_n); (A_0) \leftarrow (A_7)$ $(CY) \leftarrow (A_7)$ The content of the accumulator is rotated left one position. The low-order bit and the CY flag are both set to the value shifted out of the high-order bit position. Only the CY flag is affected	0 0 0 0 0 1 1 1	1	1		CY
Rotate Right	RRC	$(A_n) \leftarrow (A_{n+1}); (A_1) \leftarrow (A_0)$ $(CY) \leftarrow (A_0)$ The content of the accumulator is rotated right one position. The high-order bit and the CY flag are both set to the value shifted out of the low-order bit position. Only the CY flag is affected.	0 0 0 0 1 1 1 1	1	4		CY
Rotate Left Through Carry	RAL	$(A_{n+1}) \leftarrow (A_n); (CY) \leftarrow (A_7)$ $(A_0) \leftarrow (CY)$ The content of the accumulator is rotated left one position through the CY flag. The low-order bit is set equal to the CY flag and the CY flag is set to the value shifted out of the high-order bit. Only the CY flag is affected.	0 0 0 1 0 1 1 1	1	4		CY

Table F–3. Logical Group Instructions (Part 5 of 5)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
Rotate Right Through Carry	RAR	$(A_n) \leftarrow (A_{n+1})$; $(CY) \leftarrow (A_0)$ $(A_7) \leftarrow (CY)$ The content of the accumulator is rotated right one position through the CY flag. The high-order bit is set to the CY flag and the CY flag is set to the value shifted out of the low-order bit. Only the CY flag is affected.	0 0 0 1 1 1 1 1	1	4		CY
Complement Accumulator	CMA	$(A) \leftarrow (\bar{A})$ The content of the accumulator is complemented (0 bits become 1, 1 bits become 0). No flags are affected.	0 0 1 0 1 1 1 1	1	4		None
Complement Carry	CMC	$(CY) \leftarrow (\overline{CY})$ The CY flag is complemented. No other flags are affected.	0 0 1 1 1 1 1 1	1	4		CY
Set Carry	STC	$(CY) \leftarrow 1$ The CY flag is set to 1. No other flags are affected.	0 0 1 1 0 1 1 1	1	4		CY

Table F–4. Branch Group Instructions (Part 1 of 2)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing
Jump	JMP addr	(PC) ← (byte 3) (byte 2) Control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction.	1 1 0 0 0 0 1 1 Low-order address High-order address	3	10	Immediate
Conditional Jump	Jcondition addr	If (CCC), (PC) ← (byte 3) (byte 2) If the specified condition is true, control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction; otherwise, control continues sequentially.	1 1 C C C 0 1 0 Low-order address High-order address	3	10	Immediate
Call	CALL addr	((SP) – 1) ← (PCh) ((SP) – 2) ← (PCl) (SP) ← (SP) – 2 (PC) ← (byte 3) (byte 2) The high-order eight bits of the next instruction address are moved to the content of register SP. The low-order eight bits of the next instruction address are moved to the memory location whose address is 2 less than the content of register SP. The content of register SP is decremented by 2. Control is transferred to the instruction whose address is specified in byte 3 and byte 2 of the current instruction.	1 1 0 0 1 1 0 1 Low-order address High-order address	5	17	Immediate/ register indirect
Condition Call	Ccondition addr	If (CCC), ((SP) – 1) ← (PCh) ((SP) – 2) ← (PCl) (SP) ← (SP) – 2 (PC) ← (byte 3) (byte 2) If the specified condition is true, the actions specified in the CALL instruction are performed; otherwise, control continues sequentially.	1 1 C C C 1 0 0 Low-order address High-order address	3/5	11/17	Immediate/ register indirect

Table F–4. Branch Group Instructions (Part 2 of 2)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing
Return	RET	(PCl) ← ((SP)) (PCh) ← ((SP) + 1) (SP) ← (SP) + 2 The content of the memory location whose address is specified in register SP is moved to the low-order eight bits of register PC. The content of the memory location whose address is 1 more than the content of register SP is moved to the high-order eight bits of register PC. The content of register SP is incremented by 2.	1 1 0 0 1 0 0 1	3	10	Register indirect
Conditional Return	Rcondition	If (CCC), (PCl) ← ((SP)) (PCh) ← ((SP) + 1) (SP) ← (SP) + 2 If the specified condition is true, the actions specified in the RET instruction are performed; otherwise, control continues sequentially.	1 1 C C C 0 0 0	1/3	5/11	Register indirect
Restart	RST n	((SP) – 1) ← (PCH) ((SP) – 2) ← (PCL) (SP) ← (SP) – 2 (PC) ← 8 * (NNN) The high-order eight bits of the next instruction address are moved to the memory location whose address is 1 less than the content of register SP. The low-order eight bits of the next instruction address are moved to the memory location whose address is 2 less than the content of register SP. The content of register SP is decremented by 2. Control is transferred to the instruction whose address is 8 times the content of NNN.	1 1 N N N 0 0 0	3	11	Register indirect
Jump H and L Indirect, Move H and L to PC	PCHL	(PCh) ← (H) (PCl) ← (L) The content of register H is moved to the high-order eight bits of register PC. The content of register L is moved to the low-order eight bits of register PC.	1 1 1 0 1 0 0 1	1	5	Register

Table F–5. Stack, I/O, and Machine Control Group Instructions (Part 1 of 3)

Name	Mnemonic	Operation	Format	Cycles	States	Addressing	Flags
			D_7 D_0				
Push	PUSH rp	((SP) – 1) ← (rh) ((SP) – 2) ← (rl) (SP) ← (SP) – 2 The content of the high-order register of register pair rp is moved to the memory location whose address is 1 less than the content of register SP. The content of the low-order register pair rp is moved to the memory location whose address is 2 less than the content of register SP. The content of register SP is decremented by 2. NOTE: Register pair rp = SP may not be specified.	1 1 R P 0 1 0 1	3	11	Register indirect	
Push Processor Status Word	PUSH PSW	((SP) – 1) ← (A) $((SP) - 2)_0$ ← (CY), $((SP) - 2)_1$ ← 1 $((SP) - 2)_2$ ← (P), $((SP) - 2)_3$ ← 0 $((SP) - 2)_4$ ← (AC), $((SP) - 2)_5$ ← 0 $((SP) - 2)_6$ ← (Z), $((SP) - 2)_7$ ← (S) (SP) ← (SP) – 2 The content of register A is moved to the memory location whose address is 1 less than register SP. The contents of the condition flags are assembled into a processor status word and the word is moved to the memory location whose address is 2 less than the content of register SP. The content of register SP is decremented by 2.	1 1 1 1 0 1 0 1	3	11	Register indirect	

Table F–5. Stack, I/O, and Machine Control Group Instructions (Part 2 of 3)

Name	Mnemonic	Operation	Format D_7 ... D_0	Cycles	States	Addressing	Flags
Pop	POP rp	(rl) ← ((SP)) (rh) ← ((SP) + 1) (SP) ← (SP) + 2 The content of the memory location whose address is specified by the content of register SP is moved to the low-order register of register pair rp. The content of the memory location whose address is 1 more than the content of register SP is moved to the high-order register of register pair rp. The content of register SP is incremented by 2. NOTE: Register pair rp = SP may not be specified.	S Z 0 AC 0 P 1 CY	3	10	Register indirect	
Pop Processor Status Word	POP PSW	(CY) ← $((SP))_0$ (P) ← $((SP))_2$ (AC) ← $((SP))_4$ (Z) ← $((SP))_6$ (S) ← $((SP))_7$ (A) ← ((SP) + 1) (SP) ← (SP) + 2 The content of the memory location whose address is specified by the content of register SP is used to restore the condition flags. The content of the memory location whose address is 1 more than the content of register SP is moved to register A. The content of register SP is incremented by 2.	1 1 1 1 0 0 0 1	3	10	Register indirect	Z, S, P, CY, AC

Table F–5. Stack, I/O, and Machine Control Group Instructions (Part 3 of 3)

Name	Mnemonic	Operation	Format D_7 D_0	Cycles	States	Addressing	Flags
Exchange Stack Top with H and L	XTHL	(L) ↔ ((SP)) (H) ↔ ((SP) + 1 The content of the L register is exchanged with the content of the memory location whose address is specified by the content of register SP. The content of the H register is exchanged with the content of the memory location whose address is 1 more than the content of register SP.	1 1 1 0 0 0 1 1	5	18	Register indirect	
Move HL to SP	SPHL	(SP) ← (H) (L) The contents of registers H and L (16 bits) are moved to register SP.	1 1 1 1 1 0 0 1	1	5	Register	
Input	IN port	(A) ← (data) The data placed on the 8-bit bidirectional data bus by the specified port is moved to register A.	1 1 0 1 1 0 1 1 Port	3	10	Direct	
Output	OUT port	(data) ← (A) The content of register A is placed on the 8-bit bidirectional data bus for transmission to the specified port.	1 1 0 1 0 0 1 1 Port	3	10	Direct	
Enable Inturrupts	EI	The interrupt system is enabled following the execution of the next instruction.	1 1 1 1 1 0 1 1	1	4		
Disable Interrupts	DI	The interrupt system is disabled immediately following the execution of the DI instruction.	1 1 1 1 0 0 1 1	1	4		
Halt	HLT	The processor is stopped. The registers and flags are unaffected.	0 1 1 1 0 1 1 0	1	7		
No Op	NOP	No operation is performed. The registers and flags are unaffected.	0 0 0 0 0 0 0 0	1	4		

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 1 of 10)

Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
ACI $\langle B_2 \rangle$	$(A) \leftarrow (A) + \langle B_2 \rangle + (carry)$ ADD with carry	CE
ADC M	$(A) \leftarrow (A) + (M) + (carry)$ ADD with carry	8E
ADC r	$(A) \leftarrow (A) + (r) + (carry)$ Add the content of register r and the contents of the carry flip-flop to the content of the A register and place the result into Register A. (All flags affected.)	88–8D; 8F
ADD M	$(A) \leftarrow (A) + (M)$ ADD	86
ADD r	$(A) \leftarrow (A) + (r)$ Add the content of register r to the content of register A and place the result into register A. (All flags affected.)	80–85; 87
ADI $\langle B_2 \rangle$	$(A) \leftarrow (A) + \langle B_2 \rangle$ ADD	C6
ANA M	$(A) \leftarrow (A) \quad (M)$ Logical AND	A6
ANA r	$(A) \leftarrow (A) \wedge (r)$ Place the logical product of the register A and register r into register A. (Resets carry.)	A0–A5; A7
ANI $\langle B_2 \rangle$	$(A) \leftarrow (A) \wedge \langle B_2 \rangle$ Logical AND	E6
CALL $\langle B_2 \rangle$ $\langle B_3 \rangle$	$[SP - 1]\ [SP - 2] \leftarrow (PC), (SP) = (SP) - 2$ $(PC) \leftarrow \langle B_3 \rangle \langle B_2 \rangle$ Transfer the content of PC to the pushdown stack in memory addressed by the register SP. The content of SP is decremented by two. Jump unconditionally to the instruction located in memory location addressed by byte two and byte three of the instruction.	CD
CC $\langle B_2 \rangle$ $\langle B_3 \rangle$	If $(carry) = 1\ [SP - 1]\ [SP - 2] \leftarrow PC,$ $(SP) = (SP) - 2, (PC) \leftarrow \langle B_3 \rangle \langle B_2 \rangle;$ otherwise $(PC) = (PC) + 3$	DC
CM $\langle B_2 \rangle$ $\langle B_3 \rangle$	If $(sign) = 1\ [SP - 1]\ [SP - 2] \leftarrow PC,$ $(SP) = (SP) - 2, (PC) \leftarrow \langle B_3 \rangle \langle B_2 \rangle;$ otherwise $(PC) = (PC) + 3$	FC

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 2 of 10)

Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
CMA	$(A) \leftarrow (\bar{A})$ The content of accumulator is complemented. The condition flip-flops are not affected.	2F
CMC	$(carry) \leftarrow \overline{(carry)}$ The content of carry is complemented. The other condition flip-flops are not affected.	3F
CMP M	(A) – (M) COMPARE	BE
CMP r	(A) – (r) Compare the content of register A with the content of register r. The content of register A remains unchanged. The flag flip-flops are set by the result of the subtraction. Equality (A = r) is indicated by the zero flip-flop set to "1." Less than (A ⟨ r) is indicated by the carry flip-flop, set to "1."	B8–BD; BF
CNC $\langle B_2 \rangle$ $\langle B_3 \rangle$	If (carry) = 0 [SP – 1] [SP – 2] ← PC, (SP) = (SP) – 2, (PC) ← $\langle B_3 \rangle \langle B_2 \rangle$; otherwise (PC) = (PC) + 3	D4
CNZ $\langle B_2 \rangle$ $\langle B_3 \rangle$	If (zero) = 0 [SP – 1] [SP – 2] ← PC, (SP) = (SP) – 2, (PC) ← $\langle B_3 \rangle \langle B_2 \rangle$; otherwise (PC) = (PC) + 3	C4
CP $\langle B_2 \rangle$ $\langle B_3 \rangle$	If (sign) = 0 [SP – 1] [SP – 2] ← PC, (SP) = (SP) – 2, (PC) ← $\langle B_3 \rangle \langle B_2 \rangle$; otherwise (PC) = (PC) + 3	F4
CPE $\langle B_2 \rangle$ $\langle B_3 \rangle$	If (parity) = 1 [SP – 1] [SP – 2] ← PC, (SP) = (SP) – 2, (PC) ← $\langle B_3 \rangle \langle B_2 \rangle$; otherwise (PC) = (PC) + 3	EC
CPI $\langle B_2 \rangle$	(A) – $\langle B_2 \rangle$ COMPARE	FE
CPO $\langle B_2 \rangle$ $\langle B_3 \rangle$	If (parity) = 0 [SP – 1] [SP – 2] ← PC, (SP) = (SP) – 2, (PC) ← $\langle B_3 \rangle \langle B_2 \rangle$; otherwise (PC) = (PC) + 3	E4
CZ $\langle B_2 \rangle$ $\langle B_3 \rangle$	If (zero) = 1 [SP – 1] [SP – 2] ← PC, (SP) = (SP) – 2, (PC) ← $\langle B_3 \rangle \langle B_2 \rangle$; otherwise (PC) = (PC) + 3	CC

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 3 of 10)

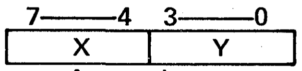
Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
DAA	Decimal Adjust Accumulator The 8-bit value in the accumulator containing the result from an arithmetic operation on decimal operands is adjusted to contain two valid BCD digits by adding a value according to the following rules:  Accumulator **If (Y $\geq$10) or (carry from bit 3) then Y = Y + 6 with carry to X digit. If (X $\geq$ 10) or (carry from bit 7) or [(Y $\geq$ 10) and (X = 9)] then X = X + 6 (which sets the carry flip-flop).** Two carry flip-flops are used for this instruction. CY_1 represents the carry from bit 3 (the fourth bit) and is accessible as a fifth flag. CY_2 is the carry from bit 7 and is the usual carry bit. All condition flip-flops are affected by this instruction.	27
DAD B	(H) (L) ← (H) (L) + (B) (C)	09
DAD D	(H) (L) ← (H) (L) + (D) (E)	19
DAD H	(H) (L) ← (H) (L) + (H) (L) (double precision shift left H and L)	29
DAD SP	(H) (L) ← (H) (L) + (SP) Add the content of register SP to the content of registers H and L and place the result into registers H and L. If the overflow is generated, the carry flip-flop is set; otherwise, the carry flip-flop is reset. The other condition flip-flops are not affected. This is useful for addressing data in the stack.	39
DCR M	[M] ← [M] – 1. The content of memory designated by registers H and L is decremented by one. All of the condition flip-flops except carry are affected by the result.	35
DCR r	(r) ← (r) – 1 The content of register r is decremented by one. All of the condition flip-flops except carry are affected by the result.	05, 0D, 15, 1D, 25, 2D, 3D
DCX B	(B) (C) ← (B) (C) – 1	0B
DCX D	(D) (E) ← (D) (E) – 1	1B
DCX H	(H) (L) ← (H) (L) – 1	2B
DCX SP	(SP) ← (SP) – 1	3B

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 4 of 10)

Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
DI	Interrupt System Disable The Interrupt Enable flip-flop (INTE) can be set or reset by using the above mentioned instructions. The INT signal will be accepted if the INTE is set. When the INT signal is accepted by the CPU, the INTE will be reset immediately. During interrupt enable or disable instruction executions, an interrupt will not be accepted.	F3
EI	Interrupt System Enable	FB
HLT	On receipt of the Halt Instruction, the activity of the processor is immediately suspended in the STOPPED state. The content of all registers and memory is unchanged and the PC has been updated.	76
IN $\langle B_2 \rangle$	(A) ← (Input Data) At T_1 time of third cycle, byte two of the instruction, which denotes the I/O device number, is sent to the I/O device through the address lines*, and the INP status information, instead of MEMR, is sent out at sync time. New data for the accumulator is loaded from the data bus when DBIN control signal is active. The condition flip-flops are not affected.	DB
INR M	[M] ← [M] + 1. The content of memory designated by registers H and L is incremented by one. All of the condition flip-flops except carry are affected by the result.	34
INR r	(r) ← (r) + 1. The content of register r is incremented by one. All the condition flip-flops except carry are affected by the result.	04, 0C, 14, 1C, 24, 2C, 3C
INX B	(B) (C) ← (B) (C) + 1 The content of register pair B and C is incremented by one. All of the condition flip-flops are not affected.	03
INX D	(D) (E) ← (D) (E) + 1	13
INX H	(H) (L) ← (H) (L) + 1 The content of register H and L is incremented by one. All of the condition flip-flops are not affected.	23
INX SP	(SP) ← (SP) + 1	33
JC $\langle B_2 \rangle$ $\langle B_3 \rangle$	If (Carry) = 1 (PC) ← $\langle B_3 \rangle \langle B_2 \rangle$ Otherwise (PC) = (PC) + 3	DA

**The device address appears on $A_7 - A_0$ and $A_{15} - A_8$.*

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 5 of 10)

Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
JM ⟨B_2⟩ ⟨B_3⟩	If (Sign) = 1 (PC) ← ⟨B_3⟩ ⟨B_2⟩ Otherwise (PC) = (PC) + 3	FA
JMP ⟨B_2⟩ ⟨B_3⟩	(PC) ← ⟨B_3⟩ ⟨B_2⟩ Jump unconditionally to the instruction located in memory location addressed by byte two and byte three.	C3
JNC ⟨B_2⟩ ⟨B_3⟩	If (Carry) = 0 (PC) ← ⟨B_3⟩ ⟨B_2⟩ Otherwise (PC) = (PC) + 3	D2
JNZ ⟨B_2⟩ ⟨B_3⟩	If (Zero) = 0 (PC) ← ⟨B_3⟩ ⟨B_2⟩ Otherwise (PC) = (PC) + 3	C2
JP ⟨B_2⟩ ⟨B_3⟩	If (Sign) = 0 (PC) ← ⟨B_3⟩ ⟨B_2⟩ Otherwise (PC) = (PC) + 3	F2
JPE ⟨B_2⟩ ⟨B_3⟩	If (Parity) = 1 (PC) ← ⟨B_3⟩ ⟨B_2⟩ Otherwise (PC) = (PC) + 3	EA
JPO ⟨B_2⟩ ⟨B_3⟩	If (Parity) = 0 (PC) ← ⟨B_3⟩ ⟨B_2⟩ Otherwise (PC) = (PC) + 3	E2
JZ ⟨B_2⟩ ⟨B_3⟩	If (Zero) = 1 (PC) ← ⟨B_3⟩ ⟨B_2⟩ Otherwise (PC) = (PC) + 3	CA
LDA ⟨B_2⟩ ⟨B_3⟩	(A) ← [⟨B_3⟩ ⟨B_2⟩] Load the accumulator with the content of the memory location addressed by byte two and byte three of the instruction.	3A
LDAX B	(A) ← [(B) (C)] Load the accumulator with the content of the memory location addressed by the content of registers B and C.	0A
LDAX D	(A) ← [(D) (E)] Load the accumulator with the content of memory location addressed by the content of register D and E.	1A
LHLD ⟨B_2⟩ ⟨B_3⟩	(L) ← [⟨B_3⟩ ⟨B_2⟩], (H) ← [⟨B_3⟩ ⟨B_2⟩ + 1] Load the registers H and L with the contents of the memory location addressed by byte two and byte three of the instruction.	2A

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 6 of 10)

Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
LXI B ⟨B_2⟩ ⟨B_3⟩	(C) ← ⟨B_2⟩; (B) ← ⟨B_3⟩ Load byte two of the instruction into C. Load byte three of the instruction into B.	01
LXI D ⟨B_2⟩ ⟨B_3⟩	(E) ← ⟨B_2⟩, (D) ← ⟨B_3⟩ Load byte two of the instruction into E. Load byte three of the instruction into D.	11
LXI H ⟨B_2⟩ ⟨B_3⟩	(L) ← ⟨B_2⟩, (H) ← ⟨B_3⟩ Load byte two of the instruction into L. Load byte three of the instruction into H.	21
LXI SP ⟨B_2⟩ ⟨B_3⟩	$(SP)_L$ ← ⟨B_2⟩, $(SP)_H$ ← ⟨B_3⟩ Load byte two of the instruction into the lower order 8-bit of the stack pointer and byte three into the higher order 8-bit of the stack pointer.	31
MVI M ⟨B_2⟩	(M) ← ⟨B_2⟩ Load byte two of the instruction into the memory location addressed by the contents of registers H and L.	36
MVI r ⟨B_2⟩	(r) ← ⟨B_2⟩ Load byte two of the instruction into register r.	06, 0E, 16, 1E, 26, 2E, 3E
MOV M, r	(M) ← (r) Load the memory location addressed by the contents of registers H and L with the content of register r.	70–77 less 76
MOV r, M	(r) ← (M) Load register r with the content of the memory location addressed by the contents of registers H and L.	46, 4E, 56, 5E, 66, 6E, 7E
MOV r_1, r_2	(r_1) ← (r_2) Load register r_1 with the content of r_2. The content of r_2 remains unchanged.	40–45; 47–4D; 4F–55; 57–5D; 5F–65; 67–6D; 78–7D
ORA M	(A) ← (A) ∨ (M) Inclusive OR	B6
ORA r	(A) ← (A) ∨ (r) Place the "inclusive-or" of the content of register A and register r into register A. (Resets carry.)	B0–B5; B7
ORI ⟨B_2⟩	(A) ← (A) ⟨B_2⟩ Inclusive OR	F6

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 7 of 10)

Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
OUT $\langle B_2 \rangle$	(Output data) ← (A) At T_1 time of the third cycle, byte two of the instruction, which denotes the I/O device number, is sent to the I/O device through the address lines*, and the OUT status information is sent out at sync time. The content of the accumulator is made available on the data bus when the $\overline{WR}$ control signal is 0.	D3
PCHL	(PC) ← (H) (L) JUMP INDIRECT	E9
POP B	(C) ← [SP], (B) ← [SP + 1], (SP) = (SP) + 2	C1
POP D	(E) ← [SP], (D) ← [SP + 1], (SP) = (SP) + 2	D1
POP H	(L) ← [SP], (H) ← [SP + 1], (SP) = (SP) + 2	E1
POP PSW	(F) ← [SP], (A) ← [SP + 1], (SP) = (SP) + 2 Restore the last values in the pushdown stack addressed by SP into A and F. The content of SP is incremented by two.	F1
PUSH B	[SP – 1] ← (B) [SP – 2] ← (C), (SP) = (SP) – 2	C5
PUSH D	[SP – 1] ← (D) [SP – 2] ← (E), (SP) = (SP) – 2	D5
PUSH H	[SP – 1] ← (H) [SP – 2] ← (L), (SP) = (SP) – 2	E5
PUSH PSW	[SP – 1] ← (A), [SP –2] ← (F), (SP) = (SP) – 2 Save the contents of A and F (5-flags) into the push-down stack addressed by the SP register. The content of SP is decremented by two. The flag word will appear as follows: D_0: CY_2 (Carry) D_1: 1 D_2: Parity (even) D_3: 0 D_4: CY_1 D_5: 0 D_6: Zero D_7: MSB (sign)	F5
RAL	A_{n+1} ← A_n, A_0 ← (carry), (carry) ← A_7 Rotate the content of Register A left one bit. Rotate the content of the carry flip-flop into A_0. Rotate A_7 into the carry flip-flop.	17

The device address appears on $A_7 - A_0$ and $A_{15} - A_8$.

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 8 of 10)

Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
RAR	$A_n \leftarrow A_{n+1}$, $A_7 \leftarrow$ (carry), (carry) $\leftarrow A_0$ Rotate the content of register A right one bit. Rotate the content of the carry flip-flop into A_7. Rotate A_0 into the carry flip-flop.	1F
RC	If (carry) = 1 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1	D8
RET	(PC) ← [SP] [SP + 1] (SP) = (SP) + 2 Return to the instruction in the memory location addressed by the last values shifted into the pushdown stack addressed by SP. The content of SP is incremented by two.	C9
RLC	$A_{n+1} \leftarrow A_n$, $A_0 \leftarrow A_7$, (carry) $\leftarrow A_7$ Rotate the content of register A left one bit. Rotate A_7 into A_0 and into the carry flip-flop.	07
RM	If (sign) =1 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1	F8
RNC	If (carry) = 0 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1	D0
RNZ	If (zero) = 0 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1	C0
RP	If (sign) = 0 (PC) ← [SP], [SP +1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1	F0
RPE	If (parity) = 1 (PC) ← [SP], [SP +1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1	E8
RPO	If (parity) = 0 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1	E0
RRC	$A_m \leftarrow A_{m+1}$, $A_7 \leftarrow A_0$, (carry) $\leftarrow A_0$ Rotate the content of register A right one bit. Rotate A_0 into A_7 and into the carry flip-flop.	0F

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 9 of 10)

Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
RST	[SP – 1] [SP – 2] ← (PC), (SP) = (SP) – 2 (PC) ← (00000000 00AAA000)	C7, CF, D7, DF, E7, EF, F7, FF
RZ	If (zero) = 1 (PC) ← [SP], [SP + 1], (SP) = (SP) + 2; otherwise (PC) = (PC) + 1	C8
SBB M	(A) ← (A) – (M) – (borrow) SUBTRACT with borrow	9E
SBB r	(A) ← (A) – (r) – (borrow) Subtract the content of register r and the content of the carry flip-flop from the content of register A and place the result into register A. (All flags affected.)	98–9D; 9F
SBI ⟨B_2⟩	(A) ← (A) – ⟨B_2⟩ – (borrow) SUBTRACT with borrow	DE
SHLD ⟨B_2⟩ ⟨B_3⟩	[⟨B_3⟩ ⟨B_2⟩] ← (L), [⟨B_3⟩ ⟨B_2⟩ + 1] ← (H) Store the contents of registers H and L into the memory location addressed by byte two and byte three of the instructions.	22
SPHL	(SP) ← (H) (L) Transfer the contents of registers H and L into register SP.	F9
STA ⟨B_2⟩ ⟨B_3⟩	[⟨B_3⟩ ⟨B_2⟩] ← (A) Store the accumulator content into the memory location addressed by byte two and byte three of the instruction.	32
STAX B	[(B) (C)] ← (A) Store the accumulator content in the memory location addressed by the content of registers B and C.	02
STAX D	[(D) (E)] ← (A) Store the accumulator content into the memory location addressed by the content of register D and E.	12
STC	(Carry) ← 1 Set the carry flip-flop to 1. The other condition flip-flops are not affected.	37
SUB M	(A) ← (A) – (M) SUBTRACT	96

Table F–6. Microprocessor Instruction Set – Alphabetical Sequence (Part 10 of 10)

Mnemonic	Description of Operation	Instruction Code (Hexadecimal)
SUB r	(A) ← (A) – (r) Subtract the content of register r from the content of register A and place the result into register A. Two's complement subtraction is used. (All flags affected.)	90–95; 97
SUI $\langle B_2 \rangle$	(A) ← (A) – $\langle B_2 \rangle$ SUBTRACT	D6
XCHG	(H) ← → (D) (E) ← → (L) Exchange the contents of registers H and L and registers D and E.	EB
XRA M	(A) ← (A) $\forall$ (M) Exclusive OR	AE
XRA r	(A) ← (A) $\forall$ (r) Place the "exclusive-or" of the content of register A and register r into register A. (Resets carry.)	A8–AD; AF
XRI $\langle B_2 \rangle$	(A) ← (A) $\langle B_2 \rangle$ Exclusive OR	EE
XTHL	(L) ← → [SP], (H) ← → [SP + 1] Exchange the contents of registers H, L and the last values in the pushdown stack addressed by register SP. The SP register itself is not changed. (SP) = (SP).	E3

Table F–7. Summary of Microprocessor Instruction Set — Hexadecimal Code Sequence (Part 1 of 8)

Hex Code	Mnemonic	Description	Binary Instruction Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
00	NOP	No-operation	0	0	0	0	0	0	0	0
01	LXI B, D16	Load immediate register pair B & C	0	0	0	0	0	0	0	1
02	STAX B	Store A indirect	0	0	0	0	0	0	1	0
03	INX B	Increment B & C registers	0	0	0	0	0	0	1	1
04	INR B	Increment register B	0	0	0	0	0	1	0	0
05	DCR B	Decrement register B	0	0	0	0	0	1	0	1
06	MVI B, D8	Move immediate register B	0	0	0	0	0	1	1	0
07	RLC	Rotate A left	0	0	0	0	0	1	1	1
08		No-operation	0	0	0	0	1	0	0	0
09	DAD B	Add B & C to H & L	0	0	0	0	1	0	0	1
0A	LDAX B	Load A indirect	0	0	0	0	1	0	1	0
0B	DCX B	Decrement B & C	0	0	0	0	1	0	1	1
0C	INR C	Increment register C	0	0	0	0	1	1	0	0
0D	DCR C	Decrement register C	0	0	0	0	1	1	0	1
0E	MVI C, D8	Move immediate register C	0	0	0	0	1	1	1	0
0F	RRC	Rotate A right	0	0	0	0	1	1	1	1
10		No-operation	0	0	0	1	0	0	0	0
11	LXI D, D16	Load immediate register pair D & E	0	0	0	1	0	0	0	1
12	STAX D	Store A indirect	0	0	0	1	0	0	1	0
13	INX D	Increment D & E registers	0	0	0	1	0	0	1	1
14	INR D	Increment register D	0	0	0	1	0	1	0	0
15	DCR D	Decrement register D	0	0	0	1	0	1	0	1
16	MVI D, D8	Move immediate register D	0	0	0	1	0	1	1	0
17	RAL	Rotate A left through carry	0	0	0	1	0	1	1	1
18		No-operation	0	0	0	1	1	0	0	0
19	DAD D	Add D & E to H & L	0	0	0	1	1	0	0	1
1A	LDAX D	Load D indirect	0	0	0	1	1	0	1	0
1B	DCX D	Decrement D & E	0	0	0	1	1	0	1	1
1C	INR E	Increment register E	0	0	0	1	1	1	0	0
1D	DCR E	Decrement register E	0	0	0	1	1	1	0	1
1E	MVI E, D8	Move immediate register E	0	0	0	1	1	1	1	0
1F	RAR	Rotate A right through carry	0	0	0	1	1	1	1	1
20		No-operation	0	0	1	0	0	0	0	0
21	LXI H, D16	Load immediate register pair H & L	0	0	1	0	0	0	0	1
22	SHLD Adr	Store H & L Direct	0	0	1	0	0	0	1	0
23	INX H	Increment H & L registers	0	0	1	0	0	0	1	1
24	INR H	Increment register H	0	0	1	0	0	1	0	0
25	DCR H	Decrement register H	0	0	1	0	0	1	0	1

Table F–7. Summary of Microprocessor Instruction Set – Hexadecimal Code Sequence (Part 2 of 8)

Hex Code	Mnemonic	Description	Binary Instruction Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
26	MVI H, D8	Move immediate register H	0	0	1	0	0	1	1	0
27	DAA	Decimal adjust A	0	0	1	0	0	1	1	1
28		No-operation	0	0	1	0	1	0	0	0
29	DAD H	Add H & L to H & L	0	0	1	0	1	0	0	1
2A	LHLD Adr	Load H & L Direct	0	0	1	0	1	0	1	0
2B	DCX H	Decrement H & L	0	0	1	0	1	0	1	1
2C	INR L	Increment register L	0	0	1	0	1	1	0	0
2D	DCR L	Decrement register L	0	0	1	0	1	1	0	1
2E	MVI L, D8	Move immediate register L	0	0	1	0	1	1	1	0
2F	CMA	Complement A	0	0	1	0	1	1	1	1
30		No-operation	0	0	1	1	0	0	0	0
31	LXI SP, D16	Load immediate stack pointer	0	0	1	1	0	0	0	1
32	STA Adr	Store A direct	0	0	1	1	0	0	1	0
33	INX SP	Increment stack pointer	0	0	1	1	0	0	1	1
34	INR M	Increment memory	0	0	1	1	0	1	0	0
35	DCR M	Decrement memory	0	0	1	1	0	1	0	1
36	MVI M, D8	Move immediate memory	0	0	1	1	0	1	1	0
37	STC	Set carry	0	0	1	1	0	1	1	1
38		No-operation	0	0	1	1	1	0	0	0
39	DAD SP	Add stack pointer to H&L	0	0	1	1	1	0	0	1
3A	LDA Adr	Load A direct	0	0	1	1	1	0	1	0
3B	DCX SP	Decrement stack pointer	0	0	1	1	1	0	1	1
3C	INR A	Increment Accumulator	0	0	1	1	1	1	0	0
3D	DCR A	Decrement Accumulator	0	0	1	1	1	1	0	1
3E	MVI A, D8	Move immediate register A	0	0	1	1	1	1	1	0
3F	CMC	Complement carry	0	0	1	1	1	1	1	1
40	MOV B, B	No-operation	0	1	0	0	0	0	0	0
41	MOV B, C	Move register C to register B	0	1	0	0	0	0	0	1
42	MOV B, D	Move register D to register B	0	1	0	0	0	0	1	0
43	MOV B, E	Move register E to register B	0	1	0	0	0	0	1	1
44	MOV B, H	Move register H to register B	0	1	0	0	0	1	0	0
45	MOV B, L	Move register L to register B	0	1	0	0	0	1	0	1
46	MOV B, M	Move memory to register B	0	1	0	0	0	1	1	0

Table F–7. Summary of Microprocessor Instruction Set – Hexadecimal Code Sequence (Part 3 of 8)

Hex Code	Mnemonic	Description	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
47	MOV B, A	Move accumulator to register B	0	1	0	0	0	1	1	1
48	MOV C, B	Move register B to register C	0	1	0	0	1	0	0	0
49	MOV C, C	No-operation	0	1	0	0	1	0	0	1
4A	MOV C, D	Move register D to register C	0	1	0	0	1	0	1	0
4B	MOV C, E	Move register E to register C	0	1	0	0	1	0	1	1
4C	MOV C, H	Move register H to register C	0	1	0	0	1	1	0	0
4D	MOV C, L	Move register L to register C	0	1	0	0	1	1	0	1
4E	MOV C, M	Move memory to register C	0	1	0	0	1	1	1	0
4F	MOV C, A	Move accumulator to register C	0	1	0	0	1	1	1	1
50	MOV D, B	Move register B to register D	0	1	0	1	0	0	0	0
51	MOV D, C	Move register C to register D	0	1	0	1	0	0	0	1
52	MOV D, D	No-operation	0	1	0	1	0	0	1	0
53	MOV D, E	Move register E to register D	0	1	0	1	0	0	1	1
54	MOV D, H	Move register H to register D	0	1	0	1	0	1	0	0
55	MOV D, L	Move register L to register D	0	1	0	1	0	1	0	1
56	MOV D, M	Move memory to register D	0	1	0	1	0	1	1	0
57	MOV D, A	Move accumulator to register D	0	1	0	1	0	1	1	1
58	MOV E, B	Move register B to register E	0	1	0	1	1	0	0	0
59	MOV E, C	Move register C to register E	0	1	0	1	1	0	0	1
5A	MOV E, D	Move register D to register E	0	1	0	1	1	0	1	0
5B	MOV E, E	No-operation	0	1	0	1	1	0	1	1
5C	MOV E, H	Move register H to register E	0	1	0	1	1	1	0	0
5D	MOV E, L	Move register L to register E	0	1	0	1	1	1	0	1
5E	MOV E, M	Move memory to register E	0	1	0	1	1	1	1	0
5F	MOV E, A	Move accumulator to register E	0	1	0	1	1	1	1	1

Table F–7. Summary of Microprocessor Instruction Set – Hexadecimal Code Sequence (Part 4 of 8)

Hex Code	Mnemonic	Description	Binary Instruction Code							
			D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
60	MOV H, B	Move register B to register H	0	1	1	0	0	0	0	0
61	MOV H, C	Move register C to register H	0	1	1	0	0	0	0	1
62	MOV H, D	Move register D to register H	0	1	1	0	0	0	1	0
63	MOV H, E	Move register E to register H	0	1	1	0	0	0	1	1
64	MOV H, H	No-operation	0	1	1	0	0	1	0	0
65	MOV H, L	Move register L to register H	0	1	1	0	0	1	0	1
66	MOV H, M	Move memory to register H	0	1	1	0	0	1	1	0
67	MOV H, A	Move accumulator to register H	0	1	1	0	0	1	1	1
68	MOV L, B	Move register B to register L	0	1	1	0	1	0	0	0
69	MOV L, C	Move register C to register L	0	1	1	0	1	0	0	1
6A	MOV L, D	Move register D to register L	0	1	1	0	1	0	1	0
6B	MOV L, E	Move register E to register L	0	1	1	0	1	0	1	1
6C	MOV L, H	Move register H to register L	0	1	1	0	1	1	0	0
6D	MOV L, L	No-operation	0	1	1	0	1	1	0	1
6E	MOV L, M	Move memory to register L	0	1	1	0	1	1	1	0
6F	MOV L, A	Move accumulator to register L	0	1	1	0	1	1	1	1
70	MOV M, B	Move register B to memory	0	1	1	1	0	0	0	0
71	MOV M, C	Move register C to memory	0	1	1	1	0	0	0	1
72	MOV M, D	Move register D to memory	0	1	1	1	0	0	1	0
73	MOV M, E	Move register E to memory	0	1	1	1	0	0	1	1
74	MOV M, H	Move register H to memory	0	1	1	1	0	1	0	0
75	MOV M, L	Move register L to memory	0	1	1	1	0	1	0	1
76	HLT	Halt	0	1	1	1	0	1	1	0
77	MOV M, A	Move accumulator to memory	0	1	1	1	0	1	1	1
78	MOV A, B	Move register B to A	0	1	1	1	1	0	0	0
79	MOV A, C	Move register C to A	0	1	1	1	1	0	0	1
7A	MOV A, D	Move register D to A	0	1	1	1	1	0	1	0
7B	MOV A, E	Move register E to A	0	1	1	1	1	0	1	1
7C	MOV A, H	Move register H to A	0	1	1	1	1	1	0	0
7D	MOV A, L	Move register L to A	0	1	1	1	1	1	0	1

Table F–7. Summary of Microprocessor Instruction Set – Hexadecimal Code Sequence (Part 5 of 8)

Hex Code	Mnemonic	Description	Binary Instruction Code							
			D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
7E	MOV A, M	Move memory to A	0	1	1	1	1	1	1	0
7F	MOV A, A	No-operation	0	1	1	1	1	1	1	1
80	ADD B	Add register B to A	1	0	0	0	0	0	0	0
81	ADD C	Add register C to A	1	0	0	0	0	0	0	1
82	ADD D	Add register D to A	1	0	0	0	0	0	1	0
83	ADD E	Add register E to A	1	0	0	0	0	0	1	1
84	ADD H	Add register H to A	1	0	0	0	0	1	0	0
85	ADD L	Add register L to A	1	0	0	0	0	1	0	1
86	ADD M	Add memory to A	1	0	0	0	0	1	1	0
87	ADD A	Add A to A	1	0	0	0	0	1	1	1
88	ADC B	Add register B to A with carry	1	0	0	0	1	0	0	0
89	ADC C	Add register C to A with carry	1	0	0	0	1	0	0	1
8A	ADC D	Add register D to A with carry	1	0	0	0	1	0	1	0
8B	ADC E	Add register E to A with carry	1	0	0	0	1	0	1	1
8C	ADC H	Add register H to A with carry	1	0	0	0	1	1	0	0
8D	ADC L	Add register L to A with carry	1	0	0	0	1	1	0	1
8E	ADC M	Add memory to A with carry	1	0	0	0	1	1	1	0
8F	ADC A	Add A to A with carry	1	0	0	0	1	1	1	1
90	SUB B	Subtract register B from A	1	0	0	1	0	0	0	0
91	SUB C	Subtract register C from A	1	0	0	1	0	0	0	1
92	SUB D	Subtract register D from A	1	0	0	1	0	0	1	0
93	SUB E	Subtract register E from A	1	0	0	1	0	0	1	1
94	SUB H	Subtract register H from A	1	0	0	1	0	1	0	0
95	SUB L	Subtract register L from A	1	0	0	1	0	1	0	1
96	SUB M	Subtract memory from A	1	0	0	1	0	1	1	0
97	SUB A	Subtract A from A	1	0	0	1	0	1	1	1
98	SBB B	Subtract register B from A with borrow	1	0	0	1	1	0	0	0
99	SBB C	Subtract register C from A with borrow	1	0	0	1	1	0	0	1

Table F–7. Summary of Microprocessor Instruction Set – Hexadecimal Code Sequence (Part 6 of 8)

Hex Code	Mnemonic	Description	Binary Instruction Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
9A	SBB D	Subtract register D from A with borrow	1	0	0	1	1	0	1	0
9B	SBB E	Subtract register E from A with borrow	1	0	0	1	1	0	1	1
9C	SBB H	Subtract register H from A with borrow	1	0	0	1	1	1	0	0
9D	SBB L	Subtract register L from A with borrow	1	0	0	1	1	1	0	1
9E	SBB M	Subtract M from A with borrow	1	0	0	1	1	1	1	0
9F	SBB A	Subtract A from A with borrow	1	0	0	1	1	1	1	1
A0	ANA B	And register B with A	1	0	1	0	0	0	0	0
A1	ANA C	And register C with A	1	0	1	0	0	0	0	1
A2	ANA D	And register D with A	1	0	1	0	0	0	1	0
A3	ANA E	And register E with A	1	0	1	0	0	0	1	1
A4	ANA H	And register H with A	1	0	1	0	0	1	0	0
A5	ANA L	And register L with A	1	0	1	0	0	1	0	1
A6	ANA M	And memory with A	1	0	1	0	0	1	1	0
A7	ANA A	No-operation	1	0	1	0	0	1	1	1
A8	XRA B	Exclusive OR register B with A	1	0	1	0	1	0	0	0
A9	XRA C	Exclusive OR register C with A	1	0	1	0	1	0	0	1
AA	XRA D	Exclusive OR register D with A	1	0	1	0	1	0	1	0
AB	XRA E	Exclusive OR register E with A	1	0	1	0	1	0	1	1
AC	XRA H	Exclusive OR register H with A	1	0	1	0	1	1	0	0
AD	XRA L	Exclusive OR register L with A	1	0	1	0	1	1	0	1
AE	XRA M	Exclusive OR memory with A	1	0	1	0	1	1	1	0
AF	XRA A	Exclusive OR A with A	1	0	1	0	1	1	1	1
B0	ORA B	Or register B with A	1	0	1	1	0	0	0	0
B1	ORA C	Or register C with A	1	0	1	1	0	0	0	1
B2	ORA D	Or register D with A	1	0	1	1	0	0	1	0
B3	ORA E	Or register E with A	1	0	1	1	0	0	1	1
B4	ORA H	Or register H with A	1	0	1	1	0	1	0	0
B5	ORA L	Or register L with A	1	0	1	1	0	1	0	1
B6	ORA M	Or memory with A	1	0	1	1	0	1	1	0
B7	ORA A	No-operation	1	0	1	1	0	1	1	1

Table F–7. Summary of Microprocessor Instruction Set – Hexadecimal Code Sequence (Part 7 of 8)

Hex Code	Mnemonic	Description	Binary Instruction Code D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
B8	CMP B	Compare register B with A	1	0	1	1	1	0	0	0
B9	CMP C	Compare register C with A	1	0	1	1	1	0	0	1
BA	CMP D	Compare register D with A	1	0	1	1	1	0	1	0
BB	CMP E	Compare register E with A	1	0	1	1	1	0	1	1
BC	CMP H	Compare register H with A	1	0	1	1	1	1	0	0
BD	CMP L	Compare register L with A	1	0	1	1	1	1	0	1
BE	CMP M	Compare memory with A	1	0	1	1	1	1	1	0
BF	CMP A	Compare A with A	1	0	1	1	1	1	1	1
C0	RNZ	Return on no zero	1	1	0	0	0	0	0	0
C1	POP B	Pop register B & C off stack	1	1	0	0	0	0	0	1
C2	JNZ Adr	Jump on no zero	1	1	0	0	0	0	1	0
C3	JMP Adr	Jump unconditional (=CB)	1	1	0	0	0	0	1	1
C4	CNZ Adr	Call on on zero	1	1	0	0	0	1	0	0
C5	PUSH B	Push register pair B & C on stack	1	1	0	0	0	1	0	1
C6	ADI D8	Add immediate to A	1	1	0	0	0	1	1	0
C7	RST 0	Restart 0	1	1	0	0	0	1	1	1
C8	RZ	Return on zero	1	1	0	0	1	0	0	0
C9	RET Adr	Return (=D9)	1	1	0	0	1	0	0	1
CA	JZ	Jump on zero	1	1	0	0	1	0	1	0
CB	JMP Adr	Jump unconditional (=C3)	1	1	0	0	1	0	1	1
CC	CZ Adr	Call on zero	1	1	0	0	1	1	0	0
CD	CALL Adr	Call unconditional	1	1	0	0	1	1	0	1
CE	ACI D8	Add immediate to A with carry	1	1	0	0	1	1	1	0
CF	RST 1	Restart 1	1	1	0	0	1	1	1	1
D0	RNC	Return on no carry	1	1	0	1	0	0	0	0
D1	POP D	Pop register D & E off stack	1	1	0	1	0	0	0	1
D2	JNC Adr	Jump on no carry	1	1	0	1	0	0	1	0
D3	OUT D8	Output	1	1	0	1	0	0	1	1
D4	CNC Adr	Call on no carry	1	1	0	1	0	1	0	0
D5	PUSH D	Push register pair D & E on stack	1	1	0	1	0	1	0	1
D6	SUI D8	Subtract immediate from A	1	1	0	1	0	1	1	0
D7	RST 2	Restart 2	1	1	0	1	0	1	1	1
D8	RC	Return on carry	1	1	0	1	1	0	0	0
D9	RET Adr	Return (=C9)	1	1	0	1	1	0	0	1
DA	JC Adr	Jump on carry	1	1	0	1	1	0	1	0
DB	IN D8	Input	1	1	0	1	1	0	1	1
DC	CC Adr	Call on carry	1	1	0	1	1	1	0	0
DD	CALL Adr	Call unconditional (=CD)	1	1	0	1	1	1	0	1

Table F–7. Summary of Microprocessor Instruction Set – Hexadecimal Code Sequence (Part 8 of 8)

Hex Code	Mnemonic	Description	Binary Instruction Code							
			D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
DE	SBI D8	Subtract immediate from A with borrow	1	1	0	1	1	1	1	0
DF	RST 3	Restart 3	1	1	0	1	1	1	1	1
E0	RPO	Restart on parity odd	1	1	1	0	0	0	0	0
E1	POP H	Pop register pair H & L off stack	1	1	1	0	0	0	0	1
E2	JPO Adr	Jump on parity odd	1	1	1	0	0	0	1	0
E3	XTHL	Exchange top of stack H & L	1	1	1	0	0	0	1	1
E4	CPO Adr	Call on parity odd	1	1	1	0	0	1	0	0
E5	PUSH H	Push register pair H & L on stack	1	1	1	0	0	1	0	1
E6	ANI D8	And immediate with A	1	1	1	0	0	1	1	0
E7	RST 4	Restart 4	1	1	1	0	0	1	1	1
E8	RPE	Return on parity even	1	1	1	0	1	0	0	0
E9	PCHL	H & L to program counter	1	1	1	0	1	0	0	1
EA	JPE Adr	Jump on parity even	1	1	1	0	1	0	1	0
EB	XCHG	Exchange D & E, H & L registers	1	1	1	0	1	0	1	1
EC	CPE Adr	Call on parity even	1	1	1	0	1	1	0	0
ED	CALL Adr	Call unconditional*	1	1	1	0	1	1	0	1
EE	XRI D8	Exclusive OR immediate with A	1	1	1	0	1	1	1	0
EF	RST 5	Restart 5	1	1	1	0	1	1	1	1
F0	RP	Return on positive	1	1	1	1	0	0	0	0
F1	POP PSW	Pop A and flags off stack	1	1	1	1	0	0	0	1
F2	JP Adr	Jump on positive	1	1	1	1	0	0	1	0
F3	DI	Disable interrupt	1	1	1	1	0	0	1	1
F4	CP Adr	Call on positive	1	1	1	1	0	1	0	0
F5	PUSH PSW	Push A and flags on stack	1	1	1	1	0	1	0	1
F6	ORI D8	Or immediate with A	1	1	1	1	0	1	1	0
F7	RST 6	Restart 6	1	1	1	1	0	1	1	1
F8	RM	Return on minus	1	1	1	1	1	0	0	0
F9	SPHL	H & L to stack pointer	1	1	1	1	1	0	0	1
FA	JM Adr	Jump on minus	1	1	1	1	1	0	1	0
FB	EI	Enable interrupts	1	1	1	1	1	0	1	1
FC	CM Adr	Call on minus	1	1	1	1	1	1	0	0
FD	CALL Adr	Call unconditional*	1	1	1	1	1	1	0	1
FE	CPI D8	Compare immediate with A	1	1	1	1	1	1	1	0
FF	RST 7	Restart 7	1	1	1	1	1	1	1	1

**Indicated codes are functionally equivalent to hex code CD.*

Appendix G. Character-Protection Feature

G.1. GENERAL INFORMATION

The following capabilities are provided by the optional character-protection feature:

- The capability of matching the number of UNISCOPE terminal protected/unprotected fields previously available in existing host-user programs;
- Space suppression within UNISCOPE terminal unprotected fields on transmission;
- Individual UNISCOPE terminal character protection in combination with field control characters (FCCs);
- Continuation of user programmability on a concurrent basis for terminals conditioned to operate in the normal UTS 400 mode or in the UNISCOPE terminal character-protection mode.

The character-protection feature replaces the UTS 400 protected mode (FCC/PROTECT switch set to PROTECT position) with two other modes of operation: the normal UTS 400 mode and the UNISCOPE terminal character-protection mode. When the character-protection feature is part of the UTS 400, the FCC/PROTECT switch must be in the FCC position. Either mode of operation is selectable from any master, primary slave, or slave in a cluster by making an appropriate entry in the MM field of the control page (refer to 2.4). Some terminals in the cluster may operate in the normal UTS 400 mode, while others may operate in the character-protection mode.

The optional character-protection feature functions compatibly with the optional screen-bypass feature.

Many of the UTS 400 functions perform in the same manner in the character-protection mode as in the normal UTS 400 mode. Those functions that operate somewhat differently are discussed in the following paragraphs.

G.2. HOST PROCESSOR TEXT MESSAGES TO UTS 400 – CHARACTER-PROTECTION DIFFERENCES

The character-protection mode affects cursor positioning sequences involving the forward tab code (HT) and the backward tab code (ESC z), which are discussed in G.4.5.1 and G.4.5.2, respectively.

G.3. UTS 400 TEXT MESSAGES TO HOST PROCESSOR – CHARACTER-PROTECTION DIFFERENCES

The type of data sent in a message transmitted from the UTS 400 to the host processor is determined by the basic operating mode in which that UTS 400 is conditioned to operate and by the particular transmit mode specified in the control page definitions. In character-protection mode, the three transmit modes function somewhat differently than they do in normal UTS 400 mode. Refer to G.4.1.3 for a discussion of the transmit-mode functions in character-protection mode. (Refer to 4.3.2 for an explanation of the functional differences in the transmit modes when the FCC/PROTECT switch is in the FCC position and the UTS 400 is conditioned to operate in the normal UTS 400 mode.)

G.4. SCREEN CONTROL – CHARACTER-PROTECTION DIFFERENCES

G.4.1. Functions Affected by FCC/PROTECT Switch Setting

NOTE:

The FCC/PROTECT switch must be in the FCC position for the optional character-protection feature to function. If the UTS 400 is turned on with the FCC/PROTECT switch set to PROTECT when the character-protection feature is installed, the message "FCC/PROTECT SWITCH MUST BE IN FCC POSITION" will be displayed on the first line of the screen. The following discussion is based on the assumption that the FCC/PROTECT switch is in the FCC position.

In character-protection mode, the UTS 400 functions affected by the FCC/PROTECT switch setting (listed in 4.4.1.4) operate as described in the following paragraphs.

G.4.1.1. Protected-Field Definition

The protected-field definition capability (similar to the UNISCOPE terminal protected-field concept) is in effect.

The SO and SI characters are used to define UNISCOPE terminal protected fields in the host processor text message (not to define Katakana characters).

NOTE:

In the character-protection mode, the extra bit in the CRT refresh memory is used as a flag to indicate whether an individual character is protected (extra bit set) or unprotected (extra bit reset). A character with the extra bit set is referred to as a UNISCOPE terminal protected character, and a character with the extra bit reset is referred to as a UNISCOPE terminal unprotected character. Katakana characters, which normally use the extra bit in the CRT refresh memory, are not supported in the character-protection mode because that extra bit is used as the protect flag. Any Katakana characters introduced while character-protection mode is in effect will appear as Katakana characters on the screen but will be treated as UNISCOPE terminal characters by the UTS 400.

Upon receipt of the SO character, the UTS 400 displays at the current cursor location the next character following the SO as a UNISCOPE terminal protected character. Upon receipt of the SI character following the SO character in a host processor text message, the UTS 400 terminates the UNISCOPE terminal protected information and displays at the current cursor location the next character following the SI as a UNISCOPE terminal unprotected character.

G.4.1.2. FCC Field and UNISCOPE Terminal Protected Character Combination

Fields sent to and from the UTS 400 are defined by the FCC sequence, as described in 4.4.1.1 through 4.4.1.3. FCCs can be intermixed with UNISCOPE terminal protected characters in the character-protection mode; that is, UNISCOPE terminal protected and unprotected characters may become part of an FCC protected or unprotected field.

G.4.1.3. Transmit-Mode Functions

In character-protection mode, the UTS 400 transmit-mode functions are modified from the definitions given in 4.3.2. The data on the screen is transmitted according to the following conditions:

- Transmit All

  All characters are transmitted, from the SOE character (or the home position) to and including the character under the cursor. All protected and unprotected fields and their associated FCCs are also transmitted. UNISCOPE terminal protected characters are sent as normal characters not bracketed by the SO and SI characters. Nonsignificant spaces at the end of a field or line are suppressed except in the line containing the cursor.

- Transmit Variable

  All unprotected characters are transmitted, from the SOE character (or the home position) to and including the character under the cursor. All unprotected fields and their associated FCCs are also transmitted. An SUB control character (octal 032) is sent in place of each UNISCOPE terminal protected character string that occurs within an FCC unprotected field. FCC protected fields and any UNISCOPE terminal protected characters within those fields are ignored and no SUB control character is sent. Nonsignificant spaces at the end of a field or line are suppressed except in the line containing the cursor.

NOTE:

As an example of an SUB control character being substituted for a string of UNISCOPE terminal protected characters in an FCC unprotected field, consider the following situation: A string of 50 consecutive UNISCOPE terminal protected characters, the middle 30 of which are in an FCC protected field, will produce two SUB control characters, one SUB character for the first 10 UNISCOPE terminal protected characters and a second for the last 10 UNISCOPE terminal protected characters.

- Transmit Changed

  Beginning with the SOE character (or the home position) to and including the cursor, only fields that have changed and which include UNISCOPE terminal protected characters, and the FCCs associated with those changed fields, are transmitted. The UNISCOPE terminal protected characters within the changed fields are sent without being bracketed by the SO and SI codes. Nonsignificant spaces at the end of a field or line are suppressed except in the line containing the cursor.

G.4.1.4. Program Attention Key Function

Program attention keys F5 through F22 are enabled.

G.4.2. Erase Functions

G.4.2.1. Erase Unprotected Data (ESC a)

This function erases all UNISCOPE terminal and FCC unprotected characters from the cursor position to the end of the display. UNISCOPE terminal and FCC protected fields are not erased.

G.4.2.2. Erase to End of Field (ESC K)

This function erases all UNISCOPE terminal and FCC unprotected data from the cursor position to the end of the field in which the cursor is located, to the end of the display, or to one position left of the first UNISCOPE protected character in the field or display, whichever occurs first. UNISCOPE terminal protected characters and FCC protected fields are not erased.

G.4.2.3. Erase to End of Line (ESC b)

This function erases all UNISCOPE terminal and FCC unprotected characters from the cursor position to the end of the field in which the cursor is located, to the end of the line, or to the character position left of the first UNISCOPE terminal protected character in the field or line, whichever occurs first. UNISCOPE terminal and FCC protected fields are not erased.

G.4.3. Delete Functions

G.4.3.1. Delete in Line (ESC c)

If the cursor is located over a UNISCOPE terminal unprotected character or in an FCC unprotected field, this function causes the character at the cursor position to be deleted. All characters following the cursor position in that field, up to the end of the line, or one space left of the first UNISCOPE terminal protected character in the field or line, are shifted left one character location. A space appears at the end of the field, at the end of the line, or one character location left of the first UNISCOPE terminal protected character in the field or line, whichever occurs first. The cursor does not move from its original position during this function.

UNISCOPE terminal and FCC protected data and FCCs are not affected by the delete-in-line function.

G.4.3.2. Delete in Display (ESC C)

If the cursor is located over a UNISCOPE terminal unprotected character or in an FCC unprotected field, this function causes the character at the cursor position to be deleted. All characters following the cursor position in that field up to the end of the display, or one space left of the first UNISCOPE terminal protected character in the field or display, are shifted left one character location. A space appears at the end of the field, at the end of the display, or one character location left of the first UNISCOPE terminal protected character in the field or display, whichever occurs first. The cursor does not move from its original position during this function.

G.4.4. Insert Functions

G.4.4.1. Insert in Line (ESC d)

If the cursor is located over a UNISCOPE terminal unprotected character or in an FCC unprotected field, this function causes the character at the cursor position and all following characters to the end of the field, the end of the line, or one character location left of the first UNISCOPE terminal protected character in the field or line, whichever occurs first, to be shifted right one character position. A space appears at the cursor position. The character (if any) in the last character position of the field, the line, or left of the first UNISCOPE terminal protected character in the field or line is deleted.

G.4.4.2. Insert in Display (ESC D)

If the cursor is located over a UNISCOPE terminal unprotected character or in an FCC unprotected field, this function causes the character at the cursor position and all following characters to the end of the field, the end of the display, or one character location left of the first UNISCOPE terminal protected character in the field or display, whichever occurs first, to be shifted right one character position. A space appears at the cursor position. The character (if any) in the last character position of the field, the display, or left of the first UNISCOPE terminal protected character in the field or display is deleted.

G.4.5. Tabulation

G.4.5.1. Forward Tab (HT)

This function causes the cursor to move to the first tab stop following the original cursor position. The cursor stops at the first UNISCOPE terminal or FCC unprotected character position to the right of the tab stop (which may be either an FCC or a tab stop code). If there are no tab stops, the cursor moves to the home position unless that position is occupied by a UNISCOPE terminal protected character or is part of an FCC protected field. In that event, the cursor moves forward to the first UNISCOPE terminal and FCC unprotected character, or if the entire remaining display is UNISCOPE terminal and FCC protected, the cursor moves to the home position.

G.4.5.2. Backward Tab (ESC z)

This function initiates a tab operation in the opposite direction from that of a forward tab operation. The cursor stops at the first UNISCOPE terminal or FCC unprotected character position to the right of the tab stop (which may be either an FCC or a tab stop code). If there are no tab stops left of the cursor, the cursor moves to the home position unless that position is occupied by a UNISCOPE terminal protected character or is part of an FCC protected field. In that event, the cursor moves to the first UNISCOPE terminal and FCC unprotected character location, or if the entire display is UNISCOPE terminal or FCC protected, the cursor moves to the home position.

G.4.6. Screen Control Functions and Codes Summary

The screen control functions and codes used by the UTS 400 are given in Table G–1. Paragraph references indicate which functions work differently during the character-protection mode. No entry in the character-protection-mode reference column opposite a function means that function operates the same way in the character-protection mode as it does in the normal UTS 400 mode.

Table G–1. Summary of Screen Control Functions and Codes in Normal UTS 400 and Character-Protection Modes

Function	Code/Sequence	Paragraph Reference		Function	Code/Sequence	Paragraph Reference	
		U400 M*	Char.-Prot. M**			U400 M*	Char.-Prot. M**
Cursor positioning	ESC VT Y X SI	4.2.2.1		Transfer changed	ESC E	4.6.1	G.6
SOE position	ESC VT Y X NUL SI	4.3.1		Transfer variable	ESC F	4.6.1	G.6
Start of entry (SOE)	RS	4.3.1		Transfer all	ESC G	4.6.1	G.6
Cursor return (new line)	CR	4.2.2.3		Print form	ESC H	4.6.1	G.6
Cursor to home	ESC e	4.2.2.2		Print	DC2	4.6.1	
Send cursor address	ESC T	4.5.1		Print transparent	ESC DC2	4.6.1	
Erase unprotected data	ESC a	4.4.2.1	G.4.2.1	Transmit variable	DC1	4.5.6	G.5.2
Erase to end of line	ESC b	4.4.2.3	G.4.2.3	Transmit all	ESC DC1	4.5.5	G.5.1
Erase to end of field	ESC K	4.4.2.2	G.4.2.2	Transmit changed	ESC t	4.5.7	G.5.3
Erase display	ESC M	4.4.2.4		Clear changed	ESC u	4.5.8	
Erase character (space)	SP	4.4.2.5					
				Call error log	ESC P	4.5.2	
Delete in line	ESC c	4.4.3.1	G.4.3.1	Clear error log	ESC R	4.5.3	
Delete in display	ESC C	4.4.3.2	G.4.3.2				
Delete line	ESC k	4.4.3.3		Initiate confidence test	ESC Q	4.5.4	
Insert in line	ESC d	4.4.4.2	G.4.4.1				
Insert in display	ESC D	4.4.4.3	G.4.4.2	Blinking start marker	FS	4.4.7.2	
Insert line	ESC j	4.4.4.1		Blinking end marker	GS	4.4.7.3	
Line duplication	ESC y	4.4.5					
				Lock keyboard	DC4 or ESC DC4	4.5.11	
Scan left	ESC g	4.2.2.4					
Scan right	ESC h	4.2.2.4		Shift in	SI	4.4.1.4.1,	G.4.1.1,
Scan down	ESC i	4.2.2.4		Shift out	SO	4.4.1.4.2	G.4.1.3
Scan up	ESC f	4.2.2.4					
				Line feed	LF	4.6.2.2	
Forward tab	HT	4.4.6.3	G.4.5.1	Form feed	FF	4.6.2.1	
Tab stop set	ESC HT	4.4.6.1					
Backward tab	ESC z	4.4.6.4	G.4.5.2	FCC character sequence	US R C M N	4.4.1.2	
				FCC character clear	ESC w	4.4.2.6	

*Abbreviation for normal UTS 400 mode
**Abbreviation for character-protection mode.

G.5. SPECIAL HOST PROCESSOR COMMANDS AND UTS 400 RESPONSES

In the character-protection mode, various special functions of the UTS 400 can be commanded from the host processor by insertion of special control sequences in text messages. These special functions, their ASCII control codes, and the UTS 400 responses to the control codes are discussed in the following paragraphs.

G.5.1. Transmit All (ESC DC1)

This command is used by the host processor to request transmission from the UTS 400 of all FCC protected and unprotected fields, associated FCCs, and UNISCOPE terminal protected and unprotected characters. The ESC DC1 code must be sent as the last sequence before the ETX character in a text message from the host processor.

G.5.2. Transmit Variable (DC1)

This command is used by the host processor to request transmission from the UTS 400 of all FCC unprotected fields and their associated FCCs and UNISCOPE terminal unprotected characters. For each UNISCOPE terminal protected character string within an FCC unprotected field, an SUB control character (octal 032) is substituted for the string and sent in its place.

Data sent from the UTS 400 in response to the DC1 code is bracketed between the SOE character (or the home position) and the cursor.

G.5.3. Transmit Changed (ESC t)

This command is used by the host processor to request transmission from the UTS 400 of only those fields which have been changed, including UNISCOPE terminal protected characters and the FCCs associated with those fields. The ESC t code must be sent as the last sequence before the ETX character in a text message from the host processor.

G.6. PERIPHERAL CONTROL

The UTS 400 operator and the host processor can send data from the UTS 400 screen to a peripheral device by using one of the peripheral initiation commands. Table G–2 lists the command name, function, and characteristics of data transferred to the peripheral when the UTS 400 is operating in the character-protection mode. Refer to 4.6.1 for additional data about initiating peripheral control.

Table G–2. Peripheral Initiation Commands – Character-Protection Mode (Part 1 of 2)

Command Name and Code	Function	Characteristics of Data Transferred to Peripheral
Print (DC2)	Sends to peripheral all characters from SOE (or home position) to cursor. Also solicits data from a peripheral appropriately selected.	FCC sequences are not included in data stream. Spaces at end of lines are suppressed (except in line containing cursor). NOTE: UNISCOPE terminal protected characters are written as displayed, but the protected character strings are bracketed by SO and SI characters. Printers may need to be conditioned to ignore the SO and SI characters.
Print Form (ESC H)	Sends to peripheral all UNISCOPE terminal and FCC unprotected characters from SOE (or home position) to cursor. Also solicits data from a peripheral appropriately selected.	FCC and UNISCOPE terminal protected characters within FCC protected fields are replaced by spaces in data stream. UNISCOPE terminal protected characters within FCC unprotected fields are replaced by spaces in data stream. FCC sequences are not included in data stream. Spaces at end of lines are suppressed (except in line containing cursor).
Print Transparent (ESC DC2)	Sends to peripheral all characters from SOE (or home position) to cursor. Also solicits data from a peripheral appropriately selected.	Spaces are not suppressed. FCC sequences are not included in data stream. Cursor returns are suppressed in data stream. NOTE: UNISCOPE terminal protected characters are written as displayed, but the protected character strings are bracketed by SO and SI characters. Printers may need to be conditioned to ignore the SO and SI characters.
Transfer All (ESC G)	Sends to peripheral all characters from SOE (or home position) to cursor. Also solicits data from a peripheral appropriately selected.	FCC sequences and UNISCOPE terminal protected character strings bracketed by SO and SI codes are included in the data stream. Spaces at end of fields ending at an FCC are suppressed (except when a field ends with a UNISCOPE terminal protected character). Spaces at end of lines are suppressed (except in line containing cursor).

Table G–2. Peripheral Initiation Commands – Character-Protection Mode (Part 2 of 2)

Command Name and Code	Function	Characteristics of Data Transferred to Peripheral
Transfer Variable (ESC F)	Sends to peripheral only the variable (not protected) fields between SOE (or home position) and cursor. Also solicits data from a peripheral appropriately selected.	FCC sequences and UNISCOPE terminal protected character strings (which occur within FCC unprotected fields and are bracketed by SO and SI codes) are included for each field transferred. Spaces at end of fields ending at an FCC are suppressed (except when a field ends with a UNISCOPE terminal protected character). Spaces at end of lines are suppressed (except in line containing cursor).
Transfer Changed (ESC E)	Sends to peripheral only the changed fields between SOE (or home position) and cursor. Also solicits data from a peripheral appropriately selected.	FCC sequences and UNISCOPE terminal protected character strings (which occur within changed FCC unprotected fields and are bracketed by SO and SI codes) are included for each changed field transferred. Spaces at end of fields ending at an FCC are suppressed (except when a field ends with a UNISCOPE terminal protected character). Spaces at end of lines are suppressed (except in line containing cursor).

Appendix H. Programming Considerations for Magnetic Stripe Reader

H.1. GENERAL

The SPERRY UNIVAC Magnetic Stripe Reader is a read-only, nonshare-type peripheral device that reads data from the magnetic stripe on credit-card-type media and enters that data in the UTS 400. The card must be pushed through the device read station by the operator.

The magnetic stripe reader interfaces to the UTS 400 through the UTS 400 keyboard interface cable. When a master or slave station is equipped with a magnetic stripe reader, both the keyboard and reader are ready for use when the system is turned on. However, the two types of input cannot be handled simultaneously; if the keyboard is in use, the reader cannot be used, and vice versa.

The keyboard shares a microprocessor, working buffer storage, and other circuitry, including the communications interface and peripheral interfaces, with the UTS 400. The shared components are located in the master station or controller. The magnetic stripe reader has its own microprocessor, working buffer storage, and other circuitry, which are located in the keyboard housing but are considered a part of the reader. When a successful read has been completed, the card-read data is converted to the 7-bit ASCII code acceptable to the UTS 400 and passed to the UTS 400 as though it were keyboard-originated data, using the UTS 400 electronic processing components and control procedures to process the data.

The encoded data formats used by the International Air Transport Association (IATA) and the American Banking Association (ABA) are recognized by the magnetic stripe reader.* Although the reader can read either format, it cannot read both. The device must be conditioned to read the selected user format.

The magnetic stripe reader also has an automatic transmission option. With this option, data is automatically transmitted after a card read.

H.2. ABA-ENCODED DATA FORMAT

H.2.1. Coded Character Set

A binary-coded-decimal 4-bit subset with odd parity is used to encode data on the magnetic stripe of ABA-formatted cards. This character code is numeric. Refer to Table H–1 for the coded character set.

**The magnetic stripe and the data stored on the magnetic stripe must comply with the American National Standard Institute (ANSI) X4.16–1976.*

Table H–1. ABA-Coded Character Set

Bits					Row	Character
P	b_4	b_3	b_2	b_1		
1	0	0	0	0	0	0
0	0	0	0	1	1	1
0	0	0	1	0	2	2
1	0	0	1	1	3	3
0	0	1	0	0	4	4
1	0	1	0	1	5	5
1	0	1	1	0	6	6
0	0	1	1	1	7	7
0	1	0	0	0	8	8
1	1	0	0	1	9	9
1	1	0	1	0	10	:
0	1	0	1	1	11	; ⓐ
1	1	1	0	0	12	<
0	1	1	0	1	13	= ⓐ
0	1	1	1	0	14	>
1	1	1	1	1	15	? ⓐ

Legend:

ⓐ These characters have the following meanings for this application:

Row 11 ; represents "start sentinel"
Row 13 = represents "separator"
Row 15 ? represents "stop sentinel"

P = Odd parity

H.2.2. Information Format

The format of the information encoded on the magnetic stripe of an ABA-formatted card is as follows:

Start sentinel	1 character
Account number	Up to 19 characters
Separator	1 character
Discretionary data	The balance up to the maximum record length (40 characters)
Stop sentinel	1 character
Longitudinal redundancy check	1 character
Total	40 characters maximum

All the characters are displayable (including the start- and stop-sentinel characters) and are part of the maximum 40-character total.

H.2.3. Code Conversion

Because the UTS 400 uses a 7-bit ASCII code (parity is excluded) to process data and the ABA-formatted data on the card is a 4-bit code (excluding the parity bit), the ABA 4-bit code must be converted to the 7-bit ASCII code. The stripe reader accomplishes this code conversion by stripping the parity bit from each ABA character code and adding an octal value of 60 to that ABA character code to get the resultant 7-bit ASCII code.

H.2.4. Longitudinal Redundancy Check (LRC)

The magnetic stripe reader runs an LRC test on the ABA 4-bit code before it is converted to the 7-bit ASCII code. The LRC test contains the following steps:

1. The parity bit of each character code is stripped off.

2. All characters from the start sentinel to and including the stop sentinel are combined by an Exclusive OR function.

3. The result of step 2 is compared to the LRC character (minus the parity bit) that was received from the magnetic stripe of the card read.

NOTE:

The LRC character is also converted to the 7-bit ASCII code and sent to the UTS 400. An LRC test can be performed on the ASCII code received by the UTS 400 without any additional code conversion.

H.3. IATA-ENCODED DATA FORMAT

H.3.1. Coded Character Set

A 6-bit-plus-odd-parity character code is used to encode data on the magnetic stripe of IATA-formatted cards. This character code is alphanumeric. Refer to Table H–2 for the coded character set.

Table H–2. IATA-Coded Character Set

				b_6	0	0	1	1
				b_5	0	1	0	1
b_4	b_3	b_2	b_1	COLUMN / ROW	0	1	2	3
0	0	0	0	0	SP	0	@	P
0	0	0	1	1	!	1	A	Q
0	0	1	0	2	"	2	B	R
0	0	1	1	3	#	3	C	S
0	1	0	0	4	$	4	D	T
0	1	0	1	5	% [a]	5	E	U
0	1	1	0	6	&	6	F	V
0	1	1	1	7	'	7	G	W
1	0	0	0	8	(	8	H	X
1	0	0	1	9	)	9	I	Y
1	0	1	0	10	*	:	J	Z
1	0	1	1	11	+	;	K	[
1	1	0	0	12	,	<	L	\
1	1	0	1	13	−	=	M	]
1	1	1	0	14	.	>	N	^ [a]
1	1	1	1	15	/	? [a]	O	_

Legend:

[a] These characters have the following meanings for this application:

Position 0/5	%	represents "start sentinel"
1/15	?	represents "stop sentinel"
3/14	^	represents "separator"

53812

H.3.2. Information Format

The information encoded on the magnetic stripe of an IATA-formatted card can be in either of two formats, format A or B. The content of each format is as follows:

Format A

Start sentinel	1 character
Format code = "A"	1 character
Surname	
Surname separator = "/"	
Initials or first name	
Separators (when required) = "space"	2 to 26 characters
Title (when used)	
Separator (when required) = "space"	
Separator	1 character
Discretionary data	The balance up to the maximum record length (79 characters)
Stop sentinel	1 character
Longitudinal redundancy check	1 character
Total	79 characters maximum

Format B

Start sentinel	1 character
Format code = "B"	1 character
Account number	Up to 19 characters
Separator	1 character
Surname	
Surname separator = "/"	
Initials or first name	
Separators (when required) = "space"	2 to 26 characters
Title (when used)	
Separator (when required) = "space"	
Separator	1 character
Discretionary data	The balance up to the maximum record length (79 characters)
Stop sentinel	1 character
Longitudinal redundancy check	1 character
Total	79 characters maximum

All the characters are displayable (including the start- and stop-sentinel characters) and are part of the maximum 79-character total.

H.3.3. Code Conversion

Because the UTS 400 uses a 7-bit ASCII code (parity is excluded) to process data and the IATA-formatted data on the card is a 6-bit code (excluding the parity bit), the IATA 6-bit code must be converted to the 7-bit ASCII code. The stripe reader accomplishes this code conversion by stripping the parity bit from the IATA code and adding an octal value of 40 to the IATA code to get the resultant 7-bit ASCII code.

H.3.4. Longitudinal Redundancy Check (LRC)

The magnetic stripe reader runs an LRC test on the IATA 6-bit code before it is converted to the 7-bit ASCII code. The LRC test contains the following steps:

1. The parity bit of each character code is stripped off.

2. All characters from the start sentinel to and including the stop sentinel are combined by an Exclusive OR function.

3. The result of step 2 is compared to the LRC character (minus the parity bit) that was received from the magnetic stripe of the card read.

NOTE:

The LRC character is also converted to the 7-bit ASCII code and sent to the UTS 400. If an LRC test is to be performed on the data received by the UTS 400 (ASCII code), the 7-bit ASCII character codes must be converted back to their equivalent IATA character codes. This conversion can be accomplished by subtracting an octal value of 40 from each 7-bit ASCII character code. Then the LRC test can be performed again. Following the LRC test, the IATA character codes must be reconverted to their equivalent 7-bit ASCII character codes.

H.4. DATA FLOW TO THE UTS 400

The data read from the magnetic stripe of a card is stored in a working buffer of the stripe reader, where character and LRC parity are checked. If the parity check is good, the data is then converted to ASCII code. Before passing the data to the UTS 400, a DEL character (ASCII code '/,, octal code 177; see Figure 2–2) will be inserted in front of the data to identify the data transmitted from the reader. The order of the data sent to the UTS 400 is as follows:

IATA Format (A and B)	ABA Format
DEL character ('/,)	DEL character ('/,)
Start sentinel (%)	Start sentinel (;)
Data characters as converted	Data characters as converted
Stop sentinel (?)	Stop sentinel (?)
LRC character as converted	LRC character as converted
Total: 79 characters* maximum	Total: 40 characters* maximum

When the magnetic stripe reader is conditioned for automatic transmission of data after a card is read, an additional data character furnished by the stripe reader follows the data sent as described above, namely, the transmit function code (ASCII code p, octal code 160).

Once data flow begins to the UTS 400, the host processor, user program, or resident UTS 400 can determine if the data is being received correctly. Any retry capability must be a provision of the user program.

**The DEL character is not considered as one of these characters.*

H.5. STATUS REPORTING

The UTS 400 terminal system has no provisions for obtaining status codes from the magnetic stripe reader.

H.6. USER-PROGRAM INTERFACE ENTRY AND EXIT POINTS

Standard UTS 400 functions, which are performed by firmware, can be extended, enhanced, or modified by incorporating user-prepared programs. User program interfaces consist of entry points from firmware to user programs and exit points from user programs to firmware.

To gain access to the magnetic stripe reader, a user program must use the normal key entry point (NORMKB) from the UTS 400 keyboard handler.

To exit from a user program that has accepted magnetic stripe reader card-read data and return to the UTS 400 keyboard handler, the user program must use the normal key exit point (NORMFM).

Refer to 5.7 for further information regarding the entry and exit points required.

Glossary

The terms in this glossary are defined as they apply to data communications, particularly to the SPERRY UNIVAC Universal Terminal System 400 (UTS 400), its peripherals, and its communications protocol.

A

accessing

Entering mass storage files from a terminal system for purposes of reference, change, or any other file function for which the terminal is equipped.

addressing

A communications protocol method of identifying one communications line interface point and a specific terminal system at that location. Also, the same method used in identifying a specific peripheral device associated with the addressed terminal system.

ASCII

Acronym for American Standard Code for Information Interchange.

asynchronous

Literally "not synchronous." Refers to a method of timing or pacing a data transmission by starting each character with a start element and following it with one or two stop elements.

B

buffer

A place or function for the temporary holding of data. Also, a device (or software routine) used to compensate for a difference in rate of data flow, or in timing of events, when data is being transmitted from one device to another.

C

cascading

The technique of expanding the number of terminal systems at one communications interface point by connecting a terminal multiplexer to the terminal port of the primary multiplexer and then connecting single terminals to the cascaded multiplexer.

communications control procedures (protocol)

The means used to control the orderly communication of information between data communications terminals and a host processor over a data communications link.

CRT

Acronym for cathode-ray tube, the element used as a display screen in a display terminal.

D

daisy chain

A method of connecting multiple peripheral devices. The first device is connected to the peripheral interface, the next device is connected to the first device, the third device is connected to the second device, and so on.

DCM

Acronym for UNIVAC Direct Connection Module (a modem replacement).

deselection

The sequence by which peripheral devices are removed from active participation on an interface, thus precluding their involvement in data transfer.

display

The visual presentation of information either being prepared for entry into the processor storage or retrieved from processor storage.

F

FCC

Acronym for field control character, a feature of the UTS 400 that provides control of display formatting.

firmware

A program permanently resident in a processor read-only memory and providing basic machine instructions through the use of microprogramming techniques.

full-duplex

Refers to a mode of transmission in which communication takes place in two directions simultaneously.

H

half-duplex
Refers to a mode of transmission in which communication takes place in one direction at a time.

handler
A software package written for a special purpose; in data communications, it generally controls input and output between the processor and the terminal systems or other communications devices.

host or host processor
Refers to the data processing system controlling the communications environment in which the UTS 400 is working.

I

I/O
Acronym for input and output.

interactive
Refers to the process of communication between a host processor and a station in which each responds alternately to procedural formalities. (Also called "conversational.")

L

list
To print or otherwise produce a permanent representation of data.

M

modem
A contraction of modulator-demodulator. A device that modulates and demodulates signals transmitted over communications facilities.

multidrop
Refers to a communications method using two or more data communications terminals on a communications line at a single interface point.

multipoint
Refers to a communications method where two or more data communications stations interface the same communications line, each at a separate interface point.

O

offline
Refers to terminal system activity performed without access to a host processor or communications line.

online
Refers to terminal system activity on a communications line.

P

parity
An element added to the basic message or character for the purpose of checking correctness of the data transmission.

peripheral device
A device such as a printer, tape cassette system, or diskette subsystem that operates from a terminal system.

peripheral interface
The special interface in a UTS 400 master or controller designed for the peripheral devices associated with this terminal system.

point-to-point
Refers to a communications method providing an exclusive communications link between two stations. One of these stations may be the host processor.

polling
A technique for inviting a data communications terminal system to transmit status or messages at a given time.

processor
A device or group of devices, with the supporting software, capable of executing a systematic sequence of operations upon data.

protocol
See communications control procedures.

R

real time
A description applied to a computation or other data processing sequence that occurs during the actual time the related process is occurring so that the results are available for modifying or guiding the process.

S

selection

The sequence by which a particular peripheral device attached to an interface is designated as the source or destination of data.

software

The programs and routines used in the operation of data processing systems, such as assemblers, compilers, and handlers.

storage

A device into which data can be entered, in which it can be held, and from which it can be retrieved at a later time. Loosely, any device that can store data.

synchronous

Refers to a method of timing or pacing of data transmission by synchronizing the transmitting equipment and the receiving equipment with a series of synchronizing characters.

T

transfer

The conveying of data between a terminal system and one of its peripheral devices across a peripheral interface.

transmission

The conveying of data and procedural messages between a processor and a remote terminal system across a communications line.

U

UNISCOPE terminal protected character

A character that is protected from operator changes without requiring an FCC delimiter (applicable in the character-protection mode only).

Index

READER'S COMMENT SHEET

Your comments on this manual will help us improve it. Please fill in the requested information.

Name of manual: ______________________________

Manual number: UP-_____ revision number_____, including update numbers _________

Name of your company: ______________________________

Address of company: ______________________________

What is your position? ______________________________

Your level of experience: Professional ______ Knowledgeable ______ Novice ______

With what system is the equipment used? ______________________________

How do you use this manual?

As a reference source ☐	As a self-instructional text ☐
As a classroom text ☐	As ________________ ☐

Please rate this manual

As a reference source:	Good ☐	Adequate ☐	Not adequate ☐
As a text:	Good ☐	Adequate ☐	Not adequate ☐
For other uses:	Good ☐	Adequate ☐	Not adequate ☐

Add your specific comments. Give page and paragraph references where appropriate.

Thank you for your cooperation.

PLEASE SEND US YOUR COMMENTS!

We feel that this manual is an essential part of our equipment. We want to be sure it is the best, most usable manual possible. Your comments will help us achieve this goal. When you have become familiar with the manual, please fill in the other side of this form and mail the form to us. Your reply will be carefully reviewed by the persons responsible for writing and publishing this manual.

FOLD

FIRST CLASS
PERMIT NO. 2540
SALT LAKE CITY,
UTAH

BUSINESS REPLY MAIL NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

CUT

POSTAGE WILL BE PAID BY

SPERRY UNIVAC

322 NORTH 2200 WEST

SALT LAKE CITY, UTAH 84116

ATTN: MANAGER, GSD PUBLICATIONS

FOLD

NOTE:

Requests for copies of this manual and other Sperry Univac publications and for assistance in getting the most use out of your Sperry Univac equipment should be directed to your local Sperry Univac representative.