

silENT 700
electronic data terminals

Model 763/765 Memory Terminals

**Systems
Manual**



TEXAS INSTRUMENTS

Copyright 1980 by Texas Instruments Incorporated
All Rights Reserved – Printed In USA

The information and/or drawings set forth in this document and all rights in and to inventions disclosed herein and patents which might be granted thereon disclosing or employing the materials, methods, techniques or apparatus described herein are the exclusive property of Texas Instruments Incorporated.

No copies of the information or drawings shall be made without the prior consent of Texas Instruments Incorporated.

Models 763/765 Memory Terminals Systems Manual, 2203665-9701, Rev. B

Original Issue: 1 December 1978

Revised: 8 February 1980

CHANGE NOTICES				
Revision Letter	Date	ECN		Description
		Number	Level	
A	4-1-79	436780	D	Correct typographical and certain technical errors
B	2-1-80	447283	D	Update references per ECN

INTRODUCTION

This *Systems Manual* for the *Silent 700** Models 763/765 memory data terminals is intended as a detailed supplement to the *Operating Instructions*. Although the *Operating Instructions* furnished with your terminal should provide sufficient information for efficient operation of the terminal, many details concerning network operation, signal timing, and terminal setup were intentionally deleted for the sake of simplicity. This manual provides the additional information required by systems-level personnel to integrate the Model 763/765 into a communication network and prepare special operating techniques for use in the network.

The reader of this manual, unless already familiar with data terminals in general and the Models 763/765 in particular, should first read the *Operating Instructions* to learn the features and operating controls of the Models 763/765 terminals. And the reader of this manual is assumed to have knowledge of electronic data processing and its terminology.

This manual is subdivided into the following major sections:

Section I — Equipment Description discusses hardware and firmware features of the terminals and provides a list of standard features and currently available options.

Section II — Operator Commands details use of the operator commands and their relationships to systems-level operations.

Section III — Communications describes modem options, communications hardware interfaces and host-system communications procedures for the terminals plus line discipline and communications signal timing.

Section IV — Throughput and Timing explains operator communications command timing, remote command execution timing, throughput limitations and other timing considerations.

Section V — Prompt and Run Files details uses of the ASR off-line functions, the special keys, and the ADC characters in creating and implementing prompt and run files.

Section VI — Applications presents example applications of Models 763/765 for operator prompt files, run files, and remote control by a host system.

Section VII — Hardware illustrates the available national character sets and lists specifications for the Models 763/765 memory data terminals.

NOTE

The information in this manual applies to U.S.A. and international versions of both the Model 763 and the Model 765 terminals unless specifically stated otherwise. All controls and operating procedures are identical in all models. The only difference between the Model 763 and the Model 765 is their communications interfaces to remote equipment. The only differences between international and U.S.A. models are certain communications frequencies, electrical power requirements, keyboard arrangements, and character sets. Any differences in terminal models are carefully noted in this manual to prevent erroneous operation or damage to your equipment.

References

The *Models 763/765 Operating Instructions* (TI Manual No. 2203664-9701) supplied with your terminal provides basic installation and operation procedures along with practice exercises to help the new operator learn the features of the Models 763/765 memory data terminals.

A pocket-size Quick Reference Card (TI Publication No. 2203666-9701) also supplied with your terminal is intended as a source of ready information for operators with basic knowledge of the Models 763/765.

The *Models 763/765 Maintenance Manual* (TI Manual No. 2200064-9701) may be ordered from your Texas Instruments supplier. The *Maintenance Manual* provides installation and basic operating instructions, theory of operation, troubleshooting diagrams, assembly drawings and associated lists of materials, and electrical and logic schematics.

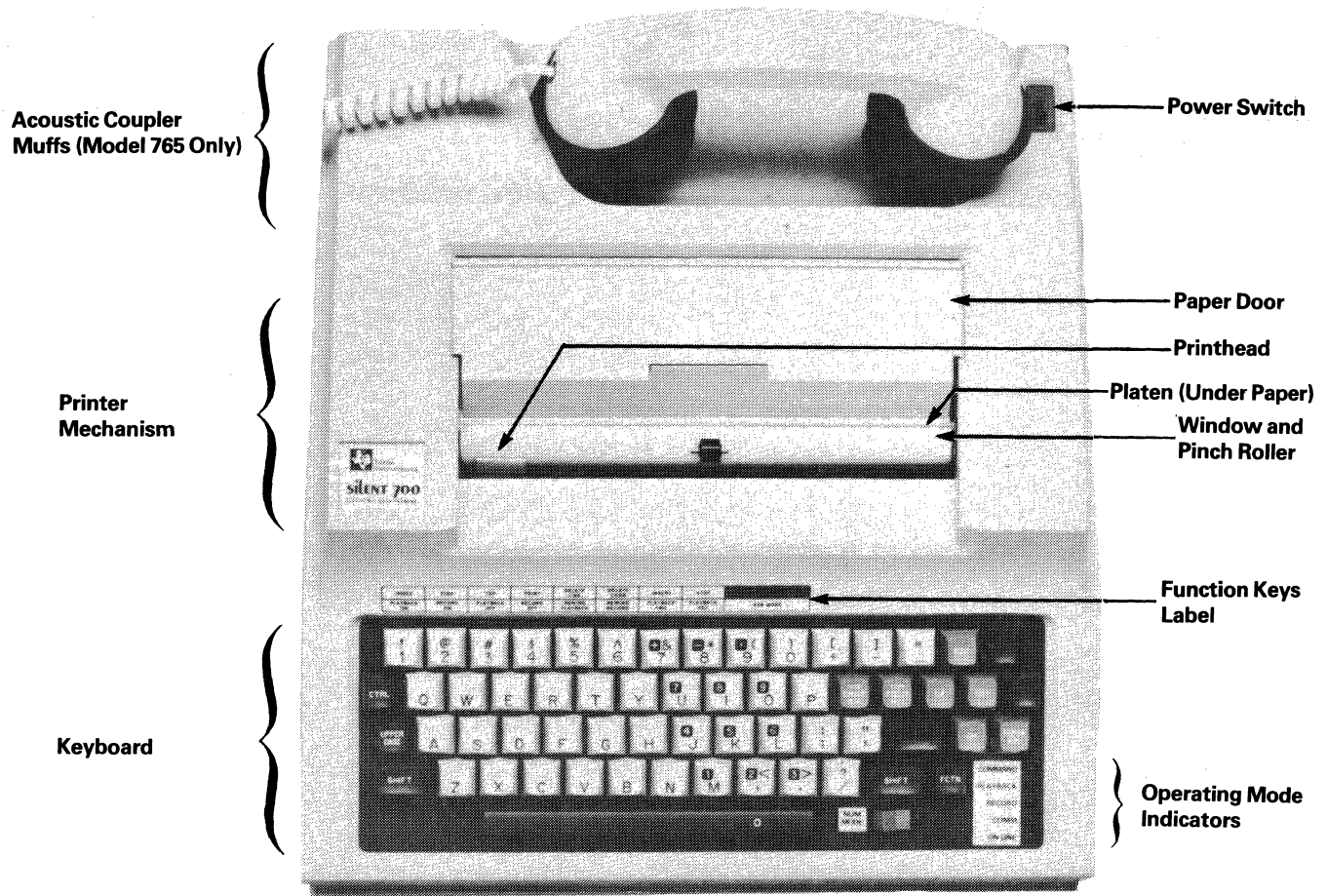
A *Self-Paced Maintenance Manual for the Models 763/765 Terminals* (TI Manual No. 2263334-9701) is available from Texas Instruments to help maintenance personnel learn fundamentals of terminal operation, theory, and maintenance.

*Trademark of Texas Instruments Incorporated

TABLE OF CONTENTS

Section	Page	Section	Page
Introduction	iii	Acoustic Coupler	25
References	iii	Carrier Detect Delay	29
I EQUIPMENT DESCRIPTION	1	EIA Line Discipline and Timing	29
Models 763/765 Hardware	1	Full Duplex	30
The Main Processor System	2	Half Duplex	30
Bubble Memory Subsystem	2	Half Duplex with Reverse Channel	30
Communications Subsystem	2	AutoABM Trigger	30
Peripheral Controller	2	ENQ Timing	30
Keyboard Printer Subsystem	2	Line Disconnect	31
Models 763/765 Firmware	2	Buffer Option Operation	31
TMS 9980 Microprocessor Firmware	2	Host System Communications with the	
TMS 8080 Microprocessor Peripheral		Models 763/765 Terminals	32
Controller Firmware	3	EDC Commands	32
Standard Features	3	IV THROUGHPUT AND TIMING	34
Optional Features	4	Operator Communications Command	
Accessories	4	Execution Timing	34
II OPERATOR COMMANDS	5	Command Execution from the	
File Utility Commands	6	Communications Line	35
File Allocation	6	Recording Data from the	
File Deallocation	8	Communications Line	35
Creation of Files	8	Acquiring Data from a Playback File	36
Deletion of Files	9	Throughput Capabilities	36
Erasing Files	9	Receive Buffer Busy	37
Locking/Freeing Files	9	V PROMPTING AND RUN FILES	38
Copying Data	9	ASR Off-Line Operation	38
Terminal Configuration Commands	11	The Function Keys	38
CHANGE Command	11	Special Keys	40
STATUS Command	12	Automatic Device Control	
Terminal Self-Diagnostic Commands	14	Characters	41
Text Editing	15	Creating Prompting Files	42
Record Locating Functions	15	Recording Constants	42
Text Modification Functions	17	Recording Variables	42
Termination Function	21	Simple Prompt/Response	42
Editing a Record	22	Other Prompt/Playback Control	
Abnormal Editor Termination	23	Characters	42
Automatic Command Execution	23	Creating RUN Files	43
III COMMUNICATIONS	24	RUN File Commands	43
Hardware Interface	24	ASR Functions in RUN Files	44
EIA Port Interface and Signals	24	Character String Constants	44
EIA Interface with an External Data		Using the RUN File	44
Set	24	Activating a Prompting File	44
Current Loop Interface for the		Reentering a RUN File from a	
Model 763	25	Prompting File	45
Optional Internal Modem for Model		Terminal Configuration RUN Files	45
763 (U.S.A. Only)	25	Data Collection within a RUN File	46

Section	Page	Section	Page
VI TYPICAL APPLICATIONS	47	VII KEYBOARDS AND CHARACTER	
Prompting Files	47	SETS	57
Sample Order Form	47	National Keyboards and Character	
Order Form Implementation	48	Sets	57
How ORDERS Form is Generated	48	Printer Character Encoding and	
RUN Files	48	Generation	57
RUN File Implementation	49		
How the ORD File Works	49	VIII SPECIFICATIONS AND	
Complete Operating Systems (To		ASSEMBLIES	62
Minimize Operator Interaction)	50	Specifications	62
How the Sample Operating System		Major Assemblies	63
Works	51		
Summary	52	Index	65
Remote Control by a Host System			
Downloading to the Terminal	53		



Model 765 Portable Memory Terminal

SECTION I

EQUIPMENT DESCRIPTION

The *Silent 700 Model 763* memory send/receive terminal and the *Model 765 Portable Memory Terminal* represent a new generation of memory terminals. The Models 763 and 765 terminals, both of which feature a typewriter-style keyboard, a thermal printer, and a non-volatile TI bubble memory system, provide important capabilities and flexibility in a compact, lightweight package.

The nonvolatile bubble memory system in the 763/765 terminals provide the user capabilities including data recording, text editing, and data storage. The nonvolatility of the memory (i.e., the memory retains the data stored in it even when power is removed) is accomplished through the use of the Texas Instruments *magnetic bubble memory system*.

The 60-key typewriter-like keyboard and the silent thermal printer facilitate use of the Models 763/765 terminals in either the KSR (keyboard send/receive) mode or in the more sophisticated and efficient ASR (automatic send/receive) mode of operation. The communications system of the 763/765 terminals may be interfaced with the majority of commercially available asynchronous modems. The Model 763 memory data terminal, designed for stationary applications, is available with either an optional internal modem (U.S. only) or a dc-current-loop interface. The Model 765 portable memory terminal is specifically designed for applications requiring a high degree of portability. The Model 765 is equipped with a built-in acoustic coupler which permits data terminal communications over standard telephone lines.

Another innovation of the Models 763/765 is the manner in which terminal configuration parameters are con-

trolled. Traditionally, terminal configuration is controlled by various switches and keys on the terminal console. The Models 763/765 terminal configuration parameters are controlled by means of *English text* operator commands which may be typed on the keyboard or programmatically executed.

The Models 763/765 terminals can best be functionally described in terms of their two basic constituents, hardware and firmware. A description of the basic hardware components and basic functions, followed by a description of the Models 763/765 firmware, is presented in the paragraphs below.

Models 763/765 Hardware

The hardware of the terminal is functionally composed of two major microprocessor systems, comprising several subsystems and a power supply as illustrated in Figure 1-1. The two major microprocessor systems and their corresponding subsystems are organized as follows:

- Main Processor System (TMS 9980 microprocessor)
 - Bubble memory subsystem
 - Communications subsystem
 - Read only memory (ROM)
 - Random access memory (RAM).
- Peripheral Controller (TMS 8080 microprocessor)
 - Keyboard/printer
 - Read only memory (ROM)
 - Random access memory (RAM).

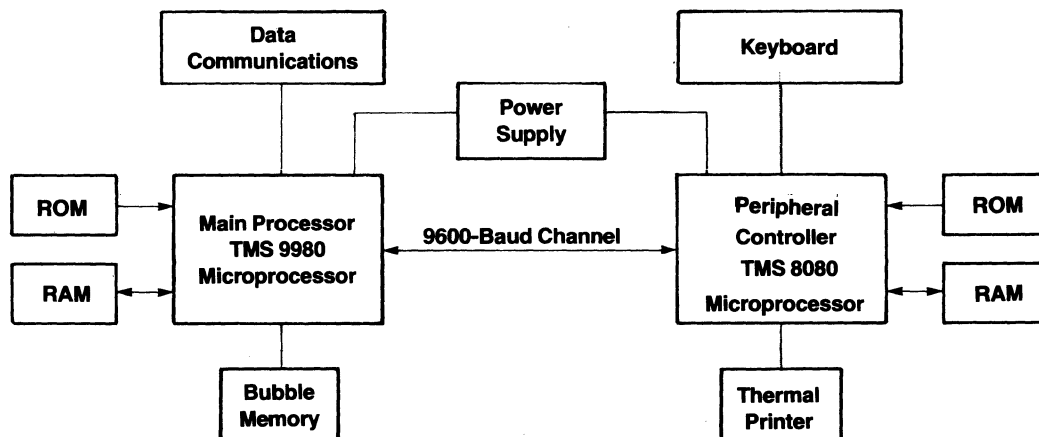


Figure 1-1. Models 763/765 Hardware Simplified Block Diagram

The Main Processor System

The heart of the main processor system is the Texas Instruments TMS 9980 microprocessor. The 9980 is responsible for interfacing the bubble memory subsystem and performing all external data communications. The main processor system contains the operating system firmware stored in read only memory (ROM). The 9980 also provides random access memory (RAM) for use in terminal operations. The main processor system performs as a *master processor*, controlling all terminal operations. The peripheral controller (TMS 8080) which serves as a *slave processor* to the TMS 9980, controls the interface to the "peripheral devices" (i.e., the keyboard and the thermal printer). The two processors communicate with one another through a 9600-baud bidirectional channel which transmits requests for service from the master processor to the slave. The channel also enables the slave processor to acknowledge requests by the master and to send it requested information.

Bubble Memory Subsystem — The bubble memory subsystem provides the capability for nonvolatile storage of 20,000 characters of data in its standard configuration. The bubble memory is expandable, in increments of 20,000 characters, up to 80,000 characters.

The bubble memory subsystem is serially oriented; that is, data exchanges are performed in chronological sequence. As a result, and because of its lower speed compared to random access memory, the bubble memory subsystem is used for mass storage of data much as a disk or tape is used in other data processing systems.

The TMS 9980 main processor system interfaces to the bubble memory subsystem via the *bubble memory controller* which controls all data exchanges.

Communications Subsystem — The *EIA communications subsystem* is capable of exchanging data through external modems or with remote devices hard wired to the terminal. The key element of communications is the TMS 9902 *asynchronous* communication controller which performs timing, data serialization, and data deserialization tasks, thus permitting the TMS 9980 main processor to control asynchronous communications. The baud rate of the TMS 9902 controller is programmable; hence the terminal baud rate can be changed simply by an instruction from the processor. Other communications parameters which are programmable include parity, parity check, duplex, interface, extended device control, auto ABM, etc.

The *acoustic coupler communications subsystem* (Model 765 only) is also controlled by the TMS 9980 main processor by means of a built-in acoustic coupler circuit which can exchange data at 110, 200 or 300 baud. The operation of the acoustic coupler is controlled by the TMS 9980 similarly to EIA communications, using the TMS 9902 asynchronous communication controller.

Peripheral Controller

The *peripheral controller* subsystem is implemented using the TMS 8080 main processor. As previously mentioned its main function is processing input/output requests from the master TMS 9980 main processor system and interfacing to the "peripheral devices": the terminal keyboard and the thermal printer.

Keyboard/Printer Subsystem — The *keyboard/printer* subsystem consists of a thermal printer and an uppercase/lowercase keyboard. The operation of both devices is indirectly controlled by the TMS 9980 main processor via the slave TMS 8080 peripheral controller. All data entered from the keyboard is detected by the TMS 8080 and is input to the TMS 9980 for processing via the 9600-baud interprocessor communications channel. In a similar manner the TMS 9980 main processor requests data to be printed by the TMS 8080 processor which, in turn, performs the thermal printer control operations.

Models 763/765 Firmware

Stored in read only memory (ROM), the *terminal firmware* is divided into two functional blocks, corresponding to each of the microprocessor systems. The main firmware block stored in the ROM subsystem of the TMS 9980 main processor contains the *operating system* and other firmware components which control operation of the terminal. The other firmware functional block stored in the TMS 8080 ROM subsystem contains the firmware responsible for controlling the peripheral devices (keyboard/printer).

TMS 9980 Microprocessor Firmware

The TMS 9980 firmware consists of of read only memory (ROM) which contains the *operating system, file management, system diagnostics, ASR emulation, device service routines, terminal utilities, and the operator communication package*. Figure 1-2 shows a block diagram of the TMS 9980 firmware.

Operating System — The 763/765 operating system is the nucleus of the terminal's firmware. Its major responsibility is allocating terminal resources to perform firmware system tasks. Examples of terminal resources include the main processor, bubble memory, and other input/output devices.

Operator Communications Package (OCP) — The OCP enables the operator to type English-text "commands" on the keyboard to be executed by the terminal. Most functions performed by the *terminal utilities* block in Figure 1-2 are activated via operator commands.

ASR Emulation Task — The ASR emulation task performs all operations in the ASR mode.

Device Service Routines — These routines serve as the interface between the operating system and the physical devices in the terminal. Devices controlled by service routines include the bubble subsystem, inter-processor communications, and the data communications ports.

System Diagnostics — The system diagnostics test the basic hardware components of the terminal without any external equipment. The system diagnostics operation is activated via an operator command.

Terminal Utilities — The terminal utilities perform a variety of functions which control operation of the terminal. The utilities available include:

- Modification of terminal configuration
- Text editing
- File utilities (e.g., file creation and deletion).

TMS 8080 Microprocessor Peripheral Controller Firmware

The TMS 8080 firmware, stored in ROM, contains the keyboard and printer control routines as well as a host of operator messages used by the terminal.

Standard Features

The following standard features are provided on the Models 763/765 memory data terminals:

- Magnetic bubble memory — nonvolatile, 20,000-character storage
- User-selectable operating configuration from the keyboard.

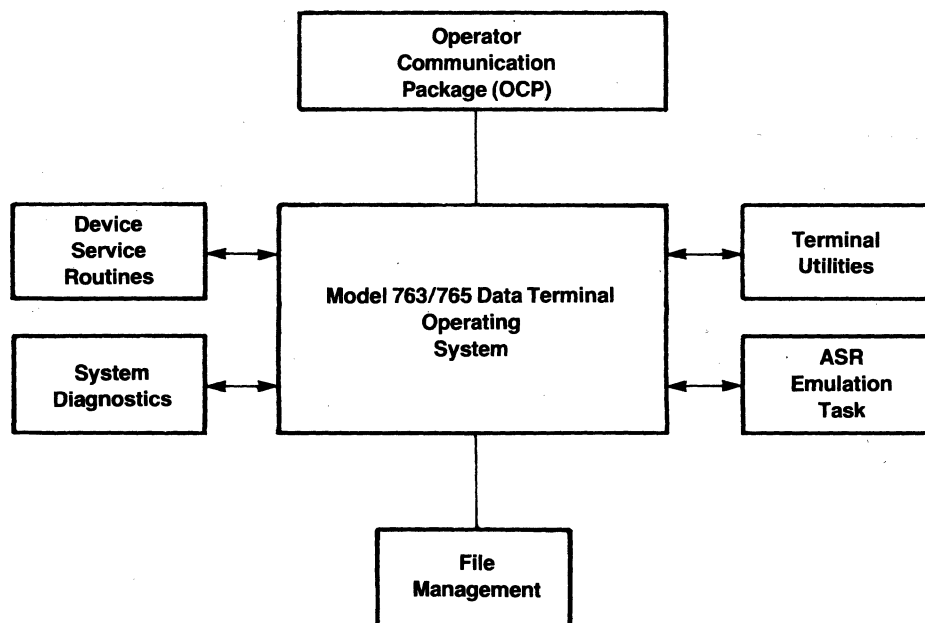


Figure 1-2. Models 763/765 Firmware Block Diagram

- Dual communications ports — external EIA (both models), internal acoustic coupler (Model 765), or dc-current loop (Model 763).
- Uppercase/lowercase keyboard
- Flexible file structure — file and record lengths are user-definable
- Two recording formats — *line* or *continuous*
- 80-character line buffer for received data
- 80-character, keyboard-selectable line buffer for transmitted data
- Automatic device control (ADC) — can be enabled to recognize DC1, 2, 3, and 4 control characters (playback on/off, record on/off)
- Extended Line Control — terminal operation can be controlled by a remote computer using ESC (escape) control codes
- Answer-back memory (ABM) — up to 34 characters may be user-defined via the keyboard for communications (secured or non-secured)
- Parity checking — enabled or disabled from the keyboard
- Horizontal tabbing (from playback files) in local mode
- Programmable key — will generate any user-defined character

- Alternate switch-selectable character sets — German, French, Swedish/Finnish, Danish/Norwegian and United Kingdom characters are provided.

Optional Features

The following options may be ordered.

- Up to 80,000 characters of nonvolatile memory in 20,000-character increments
- Built-in 113A-compatible, originate-only modem (Model 763 only) in U.S. units only
- National character keyboards and character sets (see Section VII)
- The Character Mapping Feature permits real-time substitution of characters for other characters received or transmitted by the Models 763/765. Character mapping is necessary in many timesharing applications.
- Short, 1- to 2-second carrier-detect delay
- Signal ground may be connected to chassis ground.

Accessories

The following accessories are available for the Models 763/765 memory terminals.

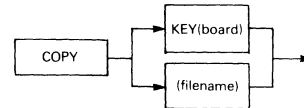
- Special interface cables to connect the Models 763/765 to other terminals and printers.

SECTION II

OPERATOR COMMANDS

This section presents detailed operating instructions and considerations of each of the operator commands. Efficient memory allocation and deallocation strategies are also discussed in this section. Instructions on accessing the COMMAND mode and entry of operator commands, as well as the syntax of each command, may be found in Section III of the *Models 763/765 Operating Instructions* (TI Manual No. 2203664-9701).

Syntax diagrams are used in this section to illustrate the format of commands. *Syntax diagrams* are graphs, always flowing from left to right, which represent the syntax rules to be followed while entering commands. An example of a syntax diagram is shown in the following figure.



The Models 763/765 provide 14 basic commands, which may be subdivided into five functional categories. In the following subsections, the operations of each of the commands in each category are discussed. Table 2-1 lists the command categories and the individual commands and abbreviations.

In this figure a portion of the COPY command syntax is illustrated. Following the flow of the diagram from left to right, we see that the COPY command must precede any of the other parameters. Then two alternate paths are available: Each alternative describes a valid syntax option which may be chosen. In this example, the KEY

TABLE 2-1. MODEL 763/765 MEMORY TERMINAL COMMANDS AND ABBREVIATIONS

CATEGORY	COMMAND	ABBREVIATION	COMMAND DESCRIPTION
File Utilities	CREATE	CF	Allocates memory space for a data file
	DELETE	(none)	Deletes entire file from the memory catalog
	ERASE	(none)	Erases the contents of a file
	LOCK	(none)	Prevents data in a file from being modified
	FREE	(none)	Releases protection provided by lock command
	COPY	CP	Copies data <i>from</i> and <i>into</i> a file
	CATALOG	CL	Lists file catalog
Communications Parameter Modification and Terminal Status Display	CHANGE	CG	Modifies communication parameters
	STATUS	ST	Displays communication parameters status
	ONLINE	ON	Places terminal in the ON-LINE mode
	OFFLINE	OFF or OF	Places terminal in the OFF-LINE mode
File Editing	EDIT	ED	Permits modification of file contents
Terminal Diagnostics	TEST	TS	Activates terminal self-test feature
Automatic Command Execution	RUN	(none)	Causes commands stored in a file to be executed

(keyboard) or the FILENAME options may be selected. The diagram then continues to flow, from left to right, to the end of the selected path. In order to simplify the syntax diagrams, the required spaces between command parameters are not illustrated.

File Utility Commands

As the name implies, file utility commands perform a number of operations which control how data is stored in the bubble memory subsystem. Using these commands the user can implement the memory allocation which is most efficient for a specific application as well as control the use of the data stored in the files.

File utility commands are performed by the *file management* system of the terminal firmware. The file utilities available include creation and deletion of files, erase file, lock file, and free file. The following brief explanation of the file management system should help you understand the individual file utility commands.

The *file management* system of the Models 763/765 is primarily responsible for storage and retrieval of all information entered into the bubble memory. Among the major responsibilities of the file management system are

- Maintenance of the file catalog
- Processing input/output requests to the bubble memory from other firmware components
- Processing file utility requests.

In this section emphasis is placed on the processing of file utilities by the file management system, as well as some considerations of its file catalog maintenance duties.

The Models 763/765 terminals are shipped from the factory with the file catalog initialized to contain no existing files. The file catalog may contain a maximum of 16 files. As files are allocated as a result of CREATE file commands, a file catalog entry is allocated to each file. Up to 16 entries are available in the catalog, corresponding to the maximum number of files that may simultaneously exist. Each catalog entry contains all information pertaining to its corresponding file. For example, each entry contains the file name, its assigned maximum size and record length, current end-of-file, file status, and other information. The file catalog is stored in a reserved area of the nonvolatile bubble memory

which may not be accessed by the user for storage of data.

A related important operation is the file management strategy for allocation and deallocation of the bubble memory storage areas. This strategy determines where new files are to be allocated in memory with respect to existing files, the availability of memory areas vacated by deleted files, and the physical structure and format of files when allocated in memory.

File Allocation

The Models 763/765 file management system comprehends the bubble memory area in which files are to be stored as a large, contiguous block of memory. When the file catalog is initialized (i.e., no files exist), the entire block of memory is available for file allocation. The size of this block depends on the amount of memory existing in the terminal. In a standard terminal the block capacity is approximately 20,000 characters. Figure 2-1.a. shows the bubble memory area as it is first initialized (the bubble memory is numbered from start to end for reference purposes). Physically these blocks correspond to the basic unit of bubble memory storage, called a *page*. Consisting of 18 bytes of data storage, each page is the indivisible bubble memory storage unit on which memory allocation is based. In other words, memory is allocated in units of entire bubble memory *pages*.

The *record*, on the other hand, is the basic unit of a file. The *record* is the unit of storage on which the user operates. File records may consist of a single page or multiple pages, depending on the designated record length. For example, if a file is created with a record length of 80 characters, each record will be allocated in units of five entire bubble memory pages. Note that even though the maximum record length requires allocation of 80 characters per record, a total of 90 characters are allocated (i.e., 5 pages \times 18 characters per page = 90 characters). This results from our previously discussed page allocation in which *partial pages* are never allocated. In the example above there are 10 characters of each record that are not used by the corresponding file even though they have been allocated. This example represents an inefficient choice of record length. The most inefficient choice, of course, is a file with record length of one character; an entire page of data storage would be allocated for each record in the file. This would mean that of the 18 bytes of data storage allocated per record, only one would be used by the file. The file would employ only 5.5 percent of the data allocated to it. In the first example (record length = 80

characters) the file would use 88.8 percent of the data allocated to it.

Therefore, to achieve maximum memory efficiency file record lengths must be a multiple of 18, in which case all data storage in the pages allocated for each record are used. Memory efficiency decreases as the record length of a file departs from a multiple of 18. Figure 2-2 illustrates a file in which 80 characters per record are allocated.

Each file is allocated by the file management system in a contiguous block of memory. The size of the block depends on the maximum number of records specified for the particular file at the time it is created. When a file is created, its contiguous block of memory is placed in that bubble memory area which follows the existing file nearest the "end" of the bubble memory area. Figure 2-1.b illustrates the allocation of a new file relative to the location of existing files. A file will never be allocated in unoccupied areas between existing files. The unoccu-

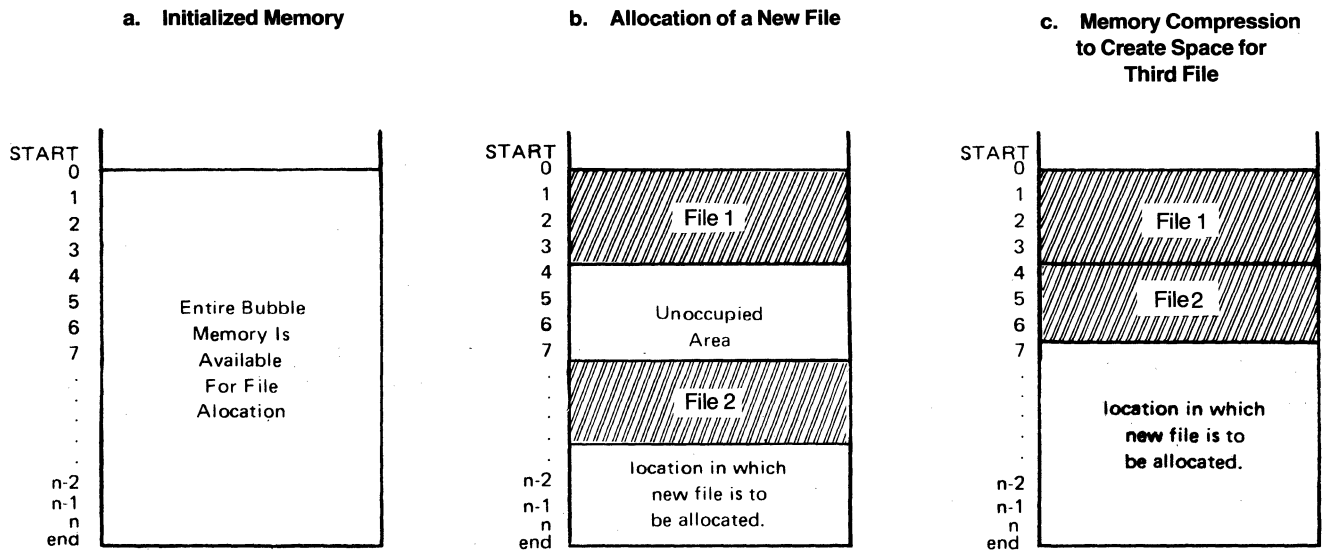
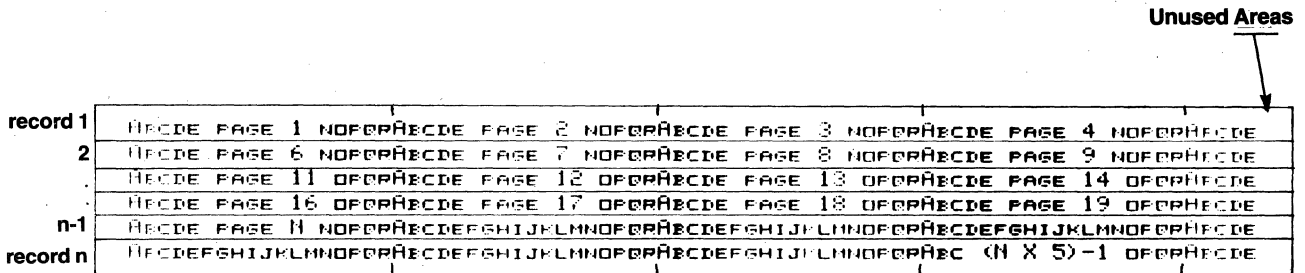


Figure 2-1. File Allocation Strategies



NOTES

1. Each page = 18 characters.
2. Each 80-character record is composed of five pages.
3. Ten characters of the last page of each record are not used.

Figure 2-2. Typical Memory Allocation in a File

pied areas result from file deletions and remain unoccupied until the memory is compressed.

If the size of the file to be created exceeds the amount of memory available for allocation of new files, the file management system will "consider" compressing the bubble memory to reclaim the unused, unoccupied areas located between existing files.

One of the duties of the file management system is to "remember" the unoccupied memory areas and their corresponding sizes. If a compression operation for allocation of a file becomes necessary, the file management system will determine if compressing the memory will reclaim enough unoccupied areas to meet the allocation requirements of the file being created. If sufficient memory can be obtained, the file management system will perform the memory compression and proceed to allocate the file. If sufficient memory cannot be reclaimed to allocate the new file, the file management system will not compress the memory, and an *insufficient memory available* error code (05) will be generated. Figure 2-1.c shows the memory allocation resulting from a memory compression of the files shown in Figure 2-1.b.

File Deallocation

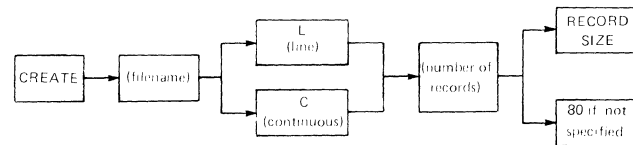
Deallocation of a file occurs when a file is deleted. Deletion of a file causes the file management system to remove the file from the file catalog. The memory occupied by the deleted file may or may not become immediately available for file allocation, depending on the location in memory of the deleted file. The *only* occasion in which the area occupied by the deleted file becomes immediately available for file allocation is the case in which the file was the last file created. If any files are located between the location of the deleted file and the "end" of memory, the area vacated by the deleted file will not become immediately available for allocation. When the vacated area does not become immediately available for allocation, the area will remain in that state until a memory compression is performed. For example, if file 2 is to be deleted from the memory allocation shown in Figure 2-1.b, the area occupied by the file would become immediately available for file allocation. However, the adjacent "unoccupied area" would remain in that state until a memory compression is performed. If file 1 in the same figure were to be deleted, the area it vacated would become "unoccupied" and unavailable for file allocation until memory is compressed.

Creation of Files

Files are created by means of the CREATE command. This command, which can be entered via the keyboard by the operator, enables the user to specify the para-

meters of the file to be created. File parameters which *must be specified* in the CREATE command include file name, maximum file size, record length, and file format.

Once the command is received by the terminal, the file management system allocates the necessary memory and prepares the file catalog entry to reflect the created file. The syntax of the CREATE command is shown below.



Several points should be considered when creating a file so as to achieve maximum memory efficiency. The first consideration is the file format chosen for the file. Data may be stored in files in the Models 763/765 in two formats: The first is the *line* format which is easy to edit, but it is rather memory inefficient since records in the file are not necessarily full of data. In *line* format lines of data (terminated by an EOL) correspond to physical records. In other words, only a single line of data is contained in each record. Figure 2-3.a illustrates a *line*-formatted file, the first record of which is occupied by a short line. As seen in the figure, more than half the data storage in the first record is wasted.

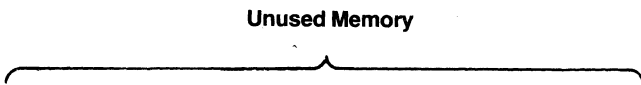
The other format in which data can be stored in a file is the *continuous* format. Using the continuous format data is stored continuously, occupying the capacity of each record. Thus, each record may contain several lines of data, each separated by an end-of-line (EOL) symbol. *Continuous* format is the most efficient since data is "packed" as much as possible in the records of the file. Figure 2-3.b shows the same data as Figure 2-3.a stored in *continuous*-formatted form.

Another consideration when creating files is the maximum size assigned to the file. Because of the fixed, contiguous structure of the files, they may not be expanded in size once created. Therefore, it is very important to select with care the maximum sizes for files used in a particular application so that the most memory efficiency is achieved.

The final consideration is the record length chosen for the files being created. Assignment of the most efficient record lengths was explained in this section, but in summary, record lengths must be assigned in multiples of 18 (i.e., 18, 36, 54 or 72 characters per record) to attain maximum memory efficiency.

a. Data Storage in the LINE Format

```
RECORD n THIS IS THE FIRST LINE OF DATA.
      n+1 THIS IS THE SECOND LINE OF DATA.
```



b. Data Storage in the CONTINUOUS Format

```
RECORD n THIS IS THE FIRST LINE OF DATA. THIS IS THE SECOND LINE OF DATA. ██████████
```



Figure 2-3. Data Storage Formats: LINE and CONTINUOUS

Deletion of Files

Deletion of a file also removes its corresponding entry from the file catalog. The data stored in the file is effectively lost since the file management system will have no record of its existence. LOCKed files cannot be deleted.

Deletion of files and creation of files should be coordinated for specific applications so that memory usage may be maximized with the minimum number of memory compression operations. For example, an application which uses some of its files for temporary storage of data may select to create these files after all of the permanent files have been created. Deletion of the temporary files, once they have been used, should then be accomplished in the reverse order in which they were created. In this scheme as temporary files are deleted, their vacated memory areas become immediately available for allocation of other files.

Erasing Files

The file management system erases a file by setting to zero the end-of-file (EOF) parameter in the file catalog entry corresponding to the erased file. In this manner the file management system quickly indicates that no data is stored in the file without having to remove the data stored in the file. LOCKed files cannot be erased.

Locking/Freeing Files

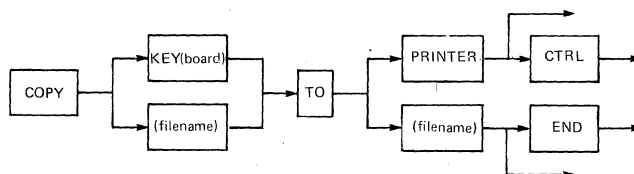
The capability to *lock* files enables the user to prevent certain files from being accidentally modified or deleted. *Freeing* of files removes the locked file protection. It is very important to note that even though files may be locked, they still will be destroyed by a system initialization (see the TEST INIT command).

Copying Data

Copying of data is enabled by the COPY command, which offers the option of copying data between the keyboard, the printer, and the memory files.

The *source* and *destination* devices are defined during entry of the COPY command. The syntax of the COPY command, showing the options available, is

Syntax:



Copying Data Between Two Files. The most widely used form of the COPY command is the copying of data from one file to another. This option may be used to copy data between files of the same or different sizes and/or record lengths, append the contents of one file to another, or change the format of the file contents (i.e., from *line* to *continuous* or vice versa).

Copying of data between files of different record lengths transfers the data from the source file to the destination file, taking in consideration their difference in record lengths. However, when both the source and the destination files are in *line* format and the destination file has a smaller record length, the data which overflows from each record in the destination file will be lost. When copying data from, or into, *continuous*-formatted files, the data is adjusted in the file records so that no data is lost during the COPY operation.

Copying of data from a *line*-formatted file to a *continuous*-formatted file causes each corresponding record in the source file to be transformed into a "line" when copied to the destination file; hence, several lines of data may be contained in a record. Each line of data is then terminated by an EOL symbol. If the record length of the destination file is less than the record length of the source file, data overflows in the destination file records will be continued in the next record. The EOL symbol then is used to terminate that line. See Figure 2-4 for an example of a copy operation between a *line* and a *continuous* format file in which the destination file has a smaller record length than that of the source file.

Copying of data from a *continuous*-formatted file to a *line*-formatted file causes each line (terminated by an EOL) in the source file to occupy a complete record in the destination *line*-formatted file. Whenever copying data into a *continuous*-formatted file, filler characters are used to right-fill the last record in the file if it is only partially filled with data. Figure 2-4 shows the form in which data is stored in the *line*-formatted file by the copy operation.

Note the filler characters at the end of the last record in the *continuous*-formatted file. If any filler characters exist in partially filled records (other than the last record in the file) upon start of the COPY operation, data will be "moved up" from the subsequent record to take the place of the filler characters. Data then will be continually moved up in the file to take the place of filler characters until the last record in the file is reached. If the last record in the file is only partially filled with data, the record will be right-filled with filler characters.

NOTE

The only reason for filler characters appearing in a partially filled record other than the last record in a file is an *abnormal text editor termination*. See the description of **File Editing** operations later in this section.

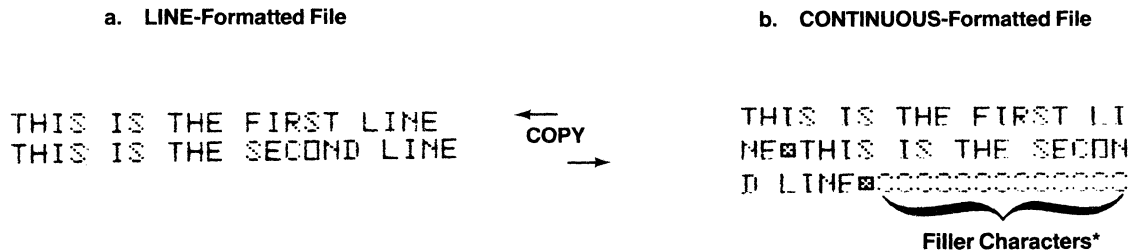
The COPY command also may be used to append the contents of one file to another file using the END option. When appending data to a *line*-formatted file, the appended data begins with the record immediately following the last existing record. When appending to a *continuous*-formatted file, any filler characters existing in the last record in the file will be removed and the appended data will begin immediately following the last character in the last existing record in the file. The appending of data follows the same format rules described for the **Copying of Data**.

Copying Data From the Keyboard Into a File. Another form of the COPY command may be used to store data being entered from the keyboard into a file. While storing data from the keyboard into a file, each line must be terminated by pressing the **SKIP** key. Once the copy operation is complete, it must be terminated by pressing the **SKIP** key and then the **ENTER** key. If the destination file is a *continuous*-formatted file, an EOL symbol will be stored each time the **SKIP** key is pressed. A new record will be started only upon reaching the maximum record length. The EOL symbol, however, is not printed. If the destination file is in *line* format, a new record will be started each time the **SKIP** key is pressed or when the maximum record length is reached.

When copying data from the keyboard to a file, the END option may be chosen. Data then is appended in the same manner as data appended from one file to another.

In general, when data is copied from the keyboard to a file, data in the record currently entered may be corrected using the **CHAR** and **FIELD** keys. Control characters entered from the keyboard will be printed.

Copying Data From a File to the Printer. When copying data from a file to the printer, the option to print control characters may be chosen; normally, control characters are executed (e.g., line feed, carriage return). When copying data from a *line*-formatted file to the



*Filler characters are printed only in the EDIT mode.

Figure 2-4. Copy Operation Data Formatting

printer, a carriage-return/line-feed is performed at the end of each record.

When copying from a *continuous*-formatted file, a carriage-return/line-feed is performed each time an EOL symbol is encountered in the file.

Terminal Configuration Commands

Terminal configuration refers to the state of a number of parameters, called *terminal configuration parameters*, which dictate the behavior of the terminal during data communications and ASR operations. The terminal configuration parameters are stored in nonvolatile bubble memory and are initialized at the factory to a set of default values. To accommodate a variety of applications which require different terminal configurations, means are provided to modify the operating configuration of the terminal. Using these means, a user can invoke the terminal configuration that best suits a particular application. Means are also provided to list the configuration parameters and their corresponding, currently assigned values. Note that since the parameters are stored in nonvolatile memory, the parameters will remain defined even if power is removed.

CHANGE Command

The means by which the values assigned to specific configuration parameters may be modified is by use of the CHANGE command. As previously mentioned, the configuration parameters control operation of the data communications and ASR operations. The general syntax of the CHANGE command is shown below.

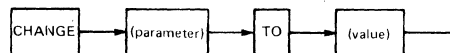


Table 2-2 lists the parameters which may be modified by the CHANGE command, along with corresponding abbreviations and values. The following two examples illustrate changing the values corresponding to the *playback file* and the *transmit EOL*.

Example 1 — Changing the Playback File:

```
▶ CHANGE PLAYBACK TO ORDERS
```

(where ORDERS is the name of an existing file).

TABLE 2-2. TERMINAL CONFIGURATION PARAMETERS

PARAMETER	ENTRY	ABBREVIATION	ACCEPTABLE VALUES
*Record file	RECORD	REC	filename
*Playback file	PLAYBACK	PLAY	filename
Auto ABM	AUTOABM		ON or OFF
ABM print	ABMPRT		ON or OFF
EOT disconnect	EOTDIS		ON or OFF
Extended device control	EDC		ON or OFF
*Programmable Key	KEY		"single character"†
DC3	DC3		ON or OFF
DC1.3	DC1.3		ON or OFF
DC2.4	DC2.4		ON or OFF
Buffer	BUFFER		ON or OFF
*Transmit EOL	XMTEOL		NL, CRLF, DC3 or "single character"†
*Receive EOL	RCVEOL		NL, CRLF or "single character"†
Transmission speed	SPEED		110,200,300,600,1200 2400,4800 or 9600 (baud)‡
Parity	PARITY		EVEN, ODD or MARK
Parity Check	PCHECK		ON or OFF
Duplex	DUPLEX		HALF, FULL, or HALFRFC
Comm. Interface	INTERFACE	PORT	INTERNAL (INT) or EIA
Answer-back memory	ABM		"up to 34 characters"†

*If no value is entered, the parameter is not defined.

†Must be entered enclosed within double quotation marks (" ")

‡Actual throughput is limited to a maximum 240 characters per second.

Example 2 — Changing the Transmit-EOL Definition

```

▶ CHANGE XMTEOL TO DC3
▶ CHANGE XMTEOL TO "6"
▶ CHANGE XMTEOL TO CRLF

```

STATUS Command

The STATUS command provides the capability to list the status of each terminal configuration parameter. The STATUS command may be used only to verify the state of the terminal configuration parameters. A typical terminal status listing is shown in Figure 2-5.

Syntax: STATUS

The parameters associated with the CHANGE and STATUS commands and their valid abbreviations (where available) are described in the following paragraphs.

INTERFACE (Port) — This parameter selects the communications port to which the terminal will communicate. *Interface* has a value of either INTERNAL (INT) or EIA. When internal is selected, the interface actually internal to the terminal will be active, either the acoustic coupler on a Model 765 or a current loop on a Model 763 (if your terminal has either of these options). When EIA is selected, the EIA interface on the rear of the terminal will be active.

SPEED — The SPEED parameter selects the speed (baud rate) at which the terminal will communicate. Valid values for speed are 110, 200, 300, 600, 1200, 2400, 4800, and 9600. Note that although the terminal will try to operate at the designated speed, the communications equipment may not be equipped to operate at that speed. The acoustic coupler can operate only up to 300 baud, and an external modem will be capable of speeds only up to 1200 baud (for some modems only). A hard-wired application using the EIA port (the bottom connector behind the flap at the rear of the terminal) will

be able to communicate at 9600 baud, but throughput to the terminal is limited to 30 characters per second when using the printer and approximately 240 characters per second when using the bubble memory (see Section IV of this manual for details on *throughput*).

PARITY — This parameter selects the parity used when transmitting and receiving data. Valid selections for the PARITY are EVEN, ODD, or MARK.

DUPLEX — This parameter selects the line discipline to be used on the communications channel. Valid selections are HALF, FULL, or HALFRFC (half duplex with reverse channel). A discussion of *line disciplines* is contained in Section III of this manual.

PCHECK — The parity check option can be set to either ON or OFF. When ON is selected, all received data will be checked for either EVEN or ODD parity, depending on the value of the PARITY parameter. If MARK parity is selected, the received data is not checked for parity regardless of the state of this option. If the PCHECK option is ON and a parity error is detected, a '?' will be substituted for the character in error. When OFF is selected, none of the received data will be checked for parity.

ABMPRT — The *print answer-back-memory* option can be set to either ON or OFF. When ON is selected, whenever the ABM is processed [by generating the HERE IS function (pressing CTRL and 1 keys) receipt of the ENQ character, or automatically sending the ABM] the ABM message will be printed unless the ABM has been secured. If this option is OFF, the ABM will not be printed.

AUTOABM — The automatic answer-back-memory option can be set to either ON or OFF. When ON is selected, automatic transmission of the ABM will occur after auto-answer is enabled. When OFF is selected, this option is inactive. For more details on the

```

▶ STATUS
LINE MODE:      INTERNAL/ 300 BAUD/ EVEN PARITY/ FULL DUPLEX/
OPTIONS ON:     EDC/ DC3/ DC1.3/ DC2.4/
OPTIONS OFF:    PCHECK/ ABMPRT/ AUTOABM/ EDTDIS/ BUFFER/
ABM:           THIS IS MY ABM
RECORD FILE:   DATA
PLAYBACK FILE: ORDERS
TRANSMIT EOL:  CRLF
RECEIVE EOL:   CRLF
KEY:
DONE

```

Figure 2-5. Sample Terminal Status Printout

AUTOABM option, refer to Section III. Auto ABM operates only if the EIA interface parameter is also selected. Auto ABM is not operable with the Model 765 using its acoustic coupler.

EOTDIS — The EOT-disconnect option can be set to either ON or OFF. When ON is selected, the terminal will disconnect itself from the communications line when an EOT character is received. When OFF is selected, the line is not monitored for the EOT character. EOTDIS is operable only if the EIA interface is also selected.

BUFFER — This option can be set to either ON or OFF. When ON is selected, all characters entered from the keyboard will be buffered (stored) until either the SKIP key is pressed or the buffer overflows (on the 81st character). When either event occurs, the buffer will be transmitted to the printer, line, or both. Before transmission, the data in the buffer can be edited using the **CHAR**, **CHAR**, **FIELD**, and **FIELD** keys. When OFF is selected, this option is disabled.

EDC — The extended device control option can be set to either ON or OFF. When ON is selected, all remote command options controlling the terminal operation from the communications line or from a playback file are enabled. The remote commands include the **ESCn** commands and the DC1 through DC4 commands. Each DC1 through DC4 command also can be controlled through some of the following parameters. The EDC option acts as a master switch for all remote operation of the terminal. When OFF is selected, no remote control option is enabled.

DC3 — This option can be set to either ON or OFF. When ON is selected (assuming EDC is also ON), a DC3 from the playback file will switch off the playback function after one more character is processed. This option also enables the multiple EDC sequences which may be stored in a playback file. When OFF, a DC3 has no effect when read from the playback file.

DC1.3 — This option can be set to either ON or OFF. When ON is selected (assuming EDC is also ON), the characters DC1 and DC3 received from the communications line will activate playback and deactivate playback respectively. If this option is OFF, the communications line will not be monitored for a DC1 or a DC3.

NOTE

If playback is ON and transmitting in *half duplex*, a DC3 cannot be received to switch off playback.

DC2.4 — This option can be set to either ON or OFF. When ON is selected (assuming EDC is also on), the characters DC2 and DC4 received from the communications line will activate record and deactivate record, respectively. If this option is OFF, the communications line will not be monitored for a DC2 or a DC4.

KEY — This option will assign any keyboard character to the programmable KEY. The syntax for this **CHANGE** command is **CHANGE KEY TO "X"**, where X is any single character enclosed in double quotation marks. Once the KEY parameter is assigned, and the key labelled KEY is pressed, the assigned character will be printed, recorded, or transmitted just as any other keyboard character would. If a noncharacter is assigned in the command [e.g., **CHANGE KEY TO (SKIP)**], the programmable KEY will be undefined and pressing the key will have no effect.

RECORD (REC) — This parameter will assign a filename to be used as the record (storage) file. The syntax for this **CHANGE** command is

CHANGE RECORD TO (filename).

The filename assigned can be any six printable characters, but must conform to filename conventions. For example, you can assign RECORD to a file named 123456, but the file management system will not let you create a file by that name since it begins with a number. A filename exceeding six characters will be truncated to six characters. The existence of the designated file will not be validated until RECORD is CHANGED to ON, at which time an error (** 88 **) will occur if the RECORD file does not exist.

PLAYBACK (PLAY) — This parameter will assign a filename to be used as the PLAYBACK file. The filename considerations for PLAYBACK are the same as those for the RECORD parameter described above.

ABM — The answer-back memory parameter defines the ABM message. The syntax of this command is **CHANGE ABM TO "up to 34 characters" (S)**. The ABM message can consist of up to 34 characters, entered within double quotation marks. To specify a double quotation (") as part of the ABM string, use two double quotation marks (""); for example,

▶ CHANGE ABM TO "PASSWORD ""PMT"" FOR 765"

This ABM message would then be **Password "PMT" for 765**. The ABM message can be secured by entering

the parameter **S** following the ABM entry. If secured, the ABM will not be printed on the terminal, regardless of the state of the ABMPRT option nor will the terminal status reveal the ABM message. If no character string is entered for the ABM in the CHANGE command, the ABM will remain undefined.

XMTEOL — This parameter defines the *transmit end-of-line*. The definitions may be a single character or a special condition. The XMTEOL character(s) is transmitted (or performed in the case of a function) when the terminal is online and communicating and the SKIP key is pressed, or when an end-of-line is detected from the playback unit (the *end-of-line* being the symbol \boxtimes in a *continuous*-formatted file, or the end of each line in a *line*-formatted file). One of the following valid options for the XMTEOL may be used.

Single Character — Any single character can be assigned to the transmit-EOL using the command **CHANGE XMTEOL TO "X"**, where X is any single character enclosed within double quotation marks.

CRLF — This option will assign the character string carriage-return/line-feed to the XMTEOL. The command syntax would be **CHANGE XMTEOL TO CRLF**. Whenever the XMTEOL is transmitted, the CR/LF control character combination will be transmitted.

NOTE

Do not use the **CR** and **LF** keys to enter this command.

NL — The command CHANGE XMTEOL TO NL will transmit a LF (line feed) control character and will perform a CR/LF when the XMTEOL is activated.

DC3 — The command CHANGE XMTEOL TO DC3 will assign the function DC3 to the XMTEOL. This function will transmit a DC3-NUL combination and will deactivate PLAYBACK whenever the XMTEOL is performed.

No Definition — The command

CHANGE XMTEOL TO

will delete any definition previously assigned to the XMTEOL. When the SKIP key is pressed, or an end-of-line is encountered, no processing will occur.

RCVEOL — This parameter defines the receive end-of-line. The definition may be a single character or a special condition. The communications line will be scanned for the receive-EOL and, when detected, the

corresponding function will be performed. If RECORD is ON, an end-of-line is recorded into the record file. One of the following valid options may be used for the receive-EOL:

Single Character — Any single character can be assigned to the RCVEOL. The command syntax is **CHANGE RCVEOL TO "X"**, where X is any character enclosed within double quotation marks. When this character is detected on received data and RECORD is ON, an end-of-line will be recorded and a CRLF will be performed if the printer is enabled.

CRLF — The command CHANGE RCVEOL TO CRLF will assign the character string CR-LF to the receive EOL. When this string is detected on received data and RECORD is ON, an end-of-line will be recorded. The terminal will act as a *state machine* when "looking" for the CR-LF sequence. When a CR is detected, the terminal will enter a "state" in which it will check the next character to sense if it is an LF. If the next character is an LF, an EOL will be recorded; if it is not, the terminal will return to its original state of "looking" for a CR. Using this scheme, the sequence CR-CR-LF will not be recognized as containing an EOL sequence.

NL — The command CHANGE RCVEOL TO NL will assign the function *new line* to the receive EOL. This function defines the character LF to be the receive EOL, with the additional property of performing a CR-LF instead of only an LF when an LF is received.

No Definition — The command CHANGE RCVEOL TO will delete any definition previously assigned to the receive EOL. Received data will not be scanned for a receive EOL.

Terminal Self-Diagnostic Commands

The self-diagnostic commands cause the terminal to test its basic hardware components without the aid of external equipment. The hardware components tested are the read only memory and the bubble memory subsystem. Execution of the TEST command does not alter the configuration or the file structure existing in the terminal.

Syntax: TEST

The TEST INIT option of the TEST command causes the terminal to reinitialize itself, returning the terminal to the state in which it was configured at the factory. In this state the terminal is configured with no files in its file catalog and the configuration parameters are configured to the *default* values listed in Table 2-3. All existing files will be destroyed when the terminal is INITIALIZED, even if the files are LOCKed.

Syntax: TEST INIT

TABLE 2-3. CHANGE COMMAND DEFAULT (OR INITIALIZED) PARAMETERS

INTERFACE (PORT)	INTERNAL
SPEED	300 baud
PARITY	even
DUPLEX	full
EDC	ON
DC3	ON
DC1.3	ON
DC2.4	ON
PCHECK	OFF
ABMPRT	OFF
AUTOABM	OFF
EOTDIS	OFF
BUFFER	OFF
XMTEOL	CRLF
RCVEOL	CRLF
ABM	no definition
RECORD	no definition
PLAYBACK	no definition
KEY	no definition

Text Editing

The contents of existing files (which are not LOCKed) may be modified using the Model 763/765 *text editor*. The terminal EDIT mode, which enables changes to the data stored in a file, is activated by means of the

EDIT (filename)

command. While in the EDIT mode the functions shown on the top row of the label above the keyboard are enabled.

The text editor is *record oriented*. That is, the contents of a file must be edited one record at a time. Specific records may be located for editing using the *record locating* functions, INDEX, FIND, TOP and PRINT. The four other editing functions are also useful to the editing task. These functions are the DELETE LINE, DELETE CHARACTER, INSERT, and STOP functions. Operation of these eight functions and other record editing operations are discussed in the following paragraphs. The basic activation and operation of the functions are described in Section III of the *Models 763/765 Operating Instructions* (TI Manual 2203664-9701) furnished with your terminal.

Files of either format may be edited, but the operation of certain functions differs somewhat from one file format to the other. The following discussion distinguishes any differences where necessary.

Record Locating Functions

The *record locating* functions offer a variety of means to specify a record to be found for modification. The record locating functions are the INDEX (F1) function, the FIND (F2) function, the TOP (F3) function, and the PRINT (F4) function. When a record locating function finds the record specified, the record is printed and is available for editing.

The record locating functions cannot be used on an empty file; any attempts will cause the terminal to signal the operator of an invalid function. In the case of the INDEX function, the signal is an audible tone; the other record locating functions print the error message *** ? ***.

While a record locating function is in progress, activation of another function will logically terminate the operations of the function in progress, and the newly selected function will be activated.

INDEX Function — F1. The INDEX function employs the *record pointer* concept to provide access to records in a file. *Records* are numbered sequentially, starting with the first record in the file. The first record in the file is assigned to *record number 1*. The record pointer may point to any record in the file, or it may point “above” the first record (i.e., record pointer = 0; record pointer = 1 points to the first record in the file). The record pointer, however, cannot go negative or exceed the number of the last existing record in the file.

The record pointer normally “points” to the record which was last accessed. For example, when a record is located for modifications, the record is found and printed out for editing. The record pointer will then point to that record. Upon initial entry into the EDIT mode, the record pointer is set to point above the first record in the file (record pointer = 0). The ability to point above the first record in the file is necessary for inserting data at the beginning of a file.

The INDEX function enables the operator to locate a record in the file relative to the record currently pointed to by the record pointer. Once the INDEX function is activated, the operator is prompted to enter a signed number from 0 through + or -999 which the terminal uses to locate the desired record relative to the current position of the record pointer. For example, if the record desired is known to be ten records before the current position of the record pointer, a -10 is entered. Similarly, if the record desired is ten records *after* the current record, a +10 is entered. The procedure for entering the signed number is discussed in Section III of the *Models 763/765 Operating Instructions*. If a 0 is entered, the current pointer position will be retained and the current record will be accessed. If the number entered places the record pointer beyond the current end-of-text, the record pointer will be set to point to the last record in the file and an ETX message will be printed. If the number entered causes the record pointer to go negative, the record pointer will be set to point above the first record in the file (record pointer = 0).

Once the relative location of the record to be accessed is entered, the record pointer will point to it. The record then will be printed, the printhead will remain at the end of the record, and the record may be edited. Each time a new record is accessed, the record pointer is correspondingly updated.

The various entries and the corresponding actions by the INDEX function may be summarized as follows:

- Entry of a signed number, followed by pressing the SKIP key, will move the record point-

er the number of records and direction requested. For example, if you want to review the six records before the current record in the file, enter a -6.

- Entry of a zero will cause the record pointer to remain at the current location.
- Pressing the SKIP key without entering a number will increment the record pointer by one record. That is, the record following the current position of the pointer will be accessed.
- When the number entered causes the record pointer to point past the end of the text, the record pointer will be set to point to the last record in the file.
- When the number entered causes the record pointer to go negative, the record pointer will be set to point above the first record in the file (i.e., record pointer = 0).
- Pressing another record locating function key (F2, F3, F4), the INSERT (F7) function key, or the STOP (F8) function key while the INDEX function is active will cause logical termination of the INDEX function and activation of the selected function.
- The INDEX function is not enabled on empty files. The audible tone is sounded to indicate an invalid operation.

FIND Function — F2. The FIND function locates a record in the file by comparing it to an operator-specified character string. The FIND function will search the file, starting from the record following the current position of the record pointer. The search for the character string will continue until its first occurrence or until the end-of-text is reached. If the specified character string is not found before the end of the text, the end-of-text (ETX) message will be printed and the record pointer will be set to point to the last record in the file.

The character string specified is limited to 30 characters in length and may be corrected during its entry using the CHAR and FIELD keys. Control characters may be included as part of the character string. For details on activation of the FIND function and entry of the search character string, see Section III of the *Models 763/765 Operating Instructions*.

When searching for a character string in *continuous*-formatted files, character strings which extend through record boundaries will be found. That is, if part of the specified string is at the end of a record while the other portion of the string continues at the beginning of the next record, the character string will be located by the FIND function (only if the string occupies no more than two records). In such case, the record pointer will be set to the record containing the last part of the string. However, character strings which are separated by an *end-of-line* (EOL) symbol will not be found.

In *line*-formatted files text will not be found across record boundaries.

As with the other record locating functions, once the desired record is located, it is printed and made available for editing.

NOTE

Since the FIND function begins the search for the specified text with the record following the current position of the record pointer, the following precaution should be taken. Unless you are sure the text to be searched for exists between the current record pointer position and the end of text, use the TOP (F3) function to return the record pointer to the first record in the file before invoking the FIND function; otherwise, the text may not be found.

Another feature of the FIND function provides multiple searches for the same text. This is useful when a particular character string occurs several places in a file. Once the first search for the string is made, subsequent searches for the same text can be made by activating the FIND function and pressing the FIELD key (SHIFT and FIELD). The previously specified text then will be printed by the terminal. Pressing the SKIP key at that point will initiate the search for the previously specified text. Once the searched for text is printed, it may be modified using the CHAR and FIELD keys. If the FIELD key causes an audible tone immediately after the FIND function is activated, it indicates that a search character string has not been entered.

The FIND function is not enabled on empty files. A *** ? *** error message will print out, indicating an invalid function.

TOP Function — F3. The TOP function causes the record pointer to point to the first record in the file. The first record will be printed and made available for editing. The TOP function is not enabled on an empty file.

PRINT Function — F4. The PRINT function will print the contents of a file, record by record, starting with the record following the current position of the record pointer. For example, if the entire file is desired, first activate the TOP function which prints the first record in the file. The record pointer is now set to this record. Then activate the PRINT function and the terminal will begin printing the contents of the entire file, starting with the second record in the file.

Once started, printing of the file will continue until one of the other *record locating* function keys (F1, F2, F3), the INSERT (F7) or STOP (F8) functions are activated, or until the end of the text is reached. If any character keys are pressed while printing is in progress, the audible tone will sound at the end of each record printed, indicating an invalid entry. While printing is in progress, invoking the PRINT function will cause the printer to stop at the end of the record currently being printed. The record then is available for editing, and the record pointer points to that record. Activating any of the other functions (except for the DELETE CHARACTER (F6) function) will cause the printer to stop at the end of the record currently being printed and the newly selected function will be activated. The DELETE CHARACTER (F6) function is not valid at this point since it is only enabled while editing a record. Once the printer stops, it may be restarted by reactivating the PRINT function.

The PRINT function is not enabled on an empty file. The *** ? *** error message will be printed, indicating an invalid function.

Text Modification Functions

Functions F5, F6, and F7 are used to edit, change, or modify the recorded text. Data may be modified character by character, or entire records may be operated on.

DELETE LINE Function — F5. The DELETE LINE function permits the operator to delete any number of existing records, starting with the current position of the record pointer. The operator may enter a positive integer from 0 through 999. Entering a 0, or pressing the SKIP key without entering a number, will abort the DELETE LINE operation and the current record will be printed and made available for editing. Entering a number from 1 through 999 will cause deletion of that number of records from the file, beginning with the current

record. The record pointer then will be set to the record immediately *following* the deleted block of records, and the record will be printed and made available for editing while the terminal is performing the deletion of specified records.

If the block of records deleted corresponds to the end of the text in the file, the record pointer will be set to the last existing record in the file. The end-of-text message (ETX) will be printed. If the entire text in a file is deleted, functions which are not enabled on empty files will be disallowed until *some* text is stored in the file.

Note that deletions of a large number of records requires a few seconds for the terminal to move existing records in the file to fill the void left by the deleted records. The data transfer rate depends on the record length of the file. For example, approximately 20 seconds is required to transfer 100 records of 80 characters each.

Activation of any function keys will not be honored until all records are transferred. However, as previously mentioned when a block of records is deleted, the record following the deleted block is printed and available for editing. Editing may be performed only on this record while the deletion process is occurring. None of the function keys which require access to another record in the file can be activated. The DELETE CHARACTER (F6) function may be used while editing the record. Access to another record in the file is prohibited until the terminal completes the transfer of records. Deletion of a block of records which encompasses the end of the text in the file will be performed very quickly since no record transfer process is required to fill the void left by the deleted records. Figure 2-6 depicts the deletion of three records in the middle of a file.

The DELETE LINE function is not enabled on empty files. The *** ? *** error message will be printed, indicating an invalid function.

DELETE CHARACTER Function — F6. The DELETE CHARACTER function is operable only while editing a record; F6 is not enabled at any other time. Character(s) in a record may be deleted using the printhead as a “pointer” to that character using F6. For example, to delete a character first position the printhead using the FIELD and/or the CHAR keys under the character to be deleted. Then, invoke the DELETE CHARACTER function to delete the character.

Upon activating the DELETE CHARACTER function, the character under which the printhead is positioned will be deleted. Deletion will be indicated by the terminal printing the DELETE CHARACTER symbol (⌘) below the character deleted. When a character is deleted, any characters to the right of the printhead will be shifted one position to the left to fill the void left by the deleted character. If the file is a *continuous*-formatted file, a filler character (⋯) will be inserted at the end of the record to fill the void caused by the left-shifted character(s).

The first time the CHAR or FIELD keys are used after deleting a character, the corrected record will be reprinted to maintain the correspondence between the printhead pointer and the pointer in memory in which the characters are stored. Once the record is reprinted, the printhead will be positioned logically. For example, if the CHAR key is pressed after deleting a character, the record will be reprinted and the printhead will execute a backspace operation relative to the position it occupied before the CHAR key was activated.

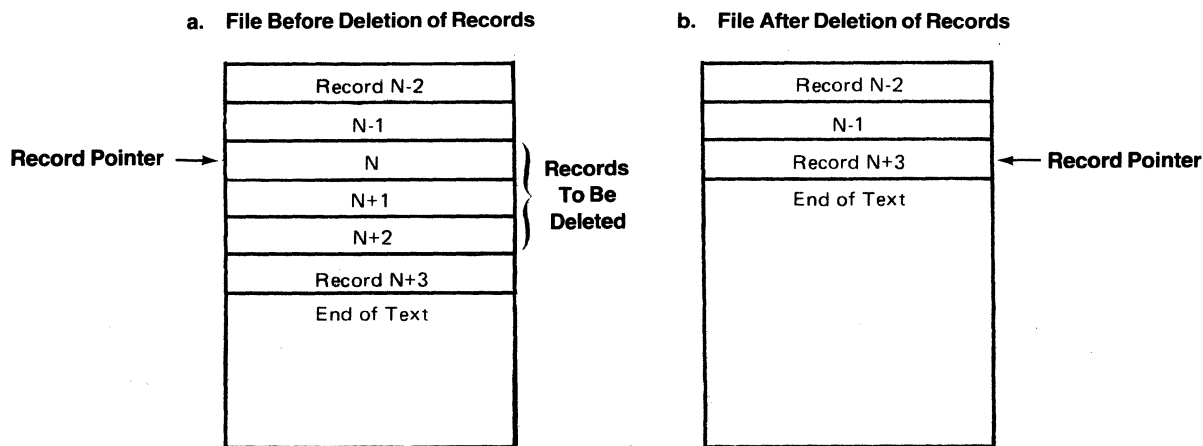


Figure 2-6. Deletion of Records in a File

INSERT Function — F7. The INSERT function permits the operator to insert characters within a record or to insert entire records within a file. When the INSERT function is activated, the “open insert” symbol (↵) is printed at the point the insertion is to begin. Upon termination of the insertion, the “close insert” symbol (␣) is printed at the point where the insertion is completed. Nested insertions are prohibited. The DELETE CHARACTER (F6) function also is prohibited during insertion of data. Attempts to activate invalid functions while inserting data will cause the audible tone to sound, notifying the operator of the invalid operation.

Generally, INSERT operations must be terminated using one of the *record locating* functions (F1-F4) or the DELETE LINE (F5) function.

Inserting Data Within A Record. Once a record is printed and available for editing, data may be inserted within that record by positioning the printhead under the first character in the record which you want to retain in memory, and then activate the INSERT (F7) function. When the INSERT function is activated while editing a record, all characters in that record at and to the right of the printhead will be saved in a temporary memory (buffer) until the insert operation is terminated.

Figure 2-7 shows a record in which an insert operation has begun. Note that the characters at and to the right of the “open insert” symbol have been saved in temporary storage.

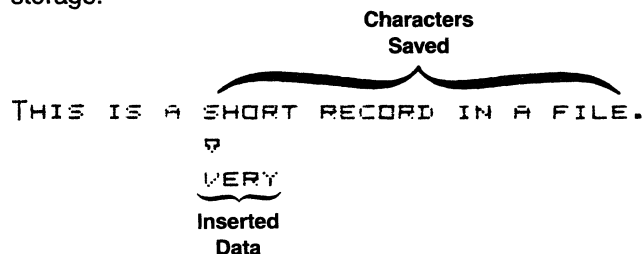


Figure 2-7. Activating an Insert Operation Within a Record

The data being inserted can be corrected using the “overstrike” feature of record editing; see **Editing a Record** later in this section. Data may be inserted in the record up to the maximum record length.

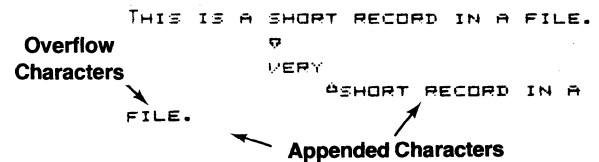
Once the insertion of data within the record is completed, the INSERT function may be terminated one of two ways. The first way to terminate is by activating any of the *record locating* functions (F1-F4). The second way is by pressing the ENTER key (the SKIP key may be used to perform this function only in *line*-formatted

files) which will cause insertion in the **current** record to terminate and permit insertion of entire records following the record just edited.

Termination of insertion within a record with a *record locating* function (F1-F4) will cause the “close insert” symbol (␣) to print, and the characters previously saved in temporary storage will be appended to the inserted data in the record. If the data inserted, plus the appended characters exceed the maximum record length of the file, the appended characters which overflow the end of the record will be placed in a newly inserted record. The appended data will be printed following the “close insert” symbol (␣). If no room remains in the file for another record, the overflow data will be lost from memory and will not be printed, and an error message (**69**) will be printed..

Figure 2-8 shows the termination of the insertion shown in Figure 2-7. Note the appending of the characters at the end of the records. Also, note that those characters that overflowed are allocated in a new record following the one just edited. The resulting text after the insert operation has been completed is also shown.

a. **Terminating the INSERT operation.**



b. **Resulting Text After Completion of INSERT Operation.**

THIS IS A VERY SHORT RECORD IN A
FILE.

Figure 2-8. Terminating an INSERT Operation Within a Record

Once the saved characters are appended, the corrected record(s) are returned to bubble memory and the *record locating* function used to terminate will be activated.

NOTE

Take care when using a record locating function (F1-F4) to terminate an INSERT operation **within an existing record**, since **inserted** characters to the right of the printhead at the time the function is activated will be truncated. This is necessary only when using the INSERT function. Also note that only inserted characters at or to the right of the printhead are discarded, not the previously existing characters.

Terminating insertion within a record by pressing the ENTER key will cause the insertion within the record to terminate but will allow the operator to continue inserting entire records following the record just edited. Upon pressing the ENTER key editing of the current record will terminate, the printer will execute a carriage-return/line-feed, and insertion of a new record may begin. For *line*-formatted files the SKIP key may be used instead of the ENTER key.

Note that the close insert symbol (␣) has not yet appeared, indicating that the insert operation is still in progress. Also note that the characters saved in temporary storage at the start of the INSERT operation are still stored until the operation is terminated.

Once the insertion of *new* records begins, the ENTER key must be pressed at the end of each new record inserted. This will cause each record inserted to be stored in bubble memory, and permit the insertion of still another new record.

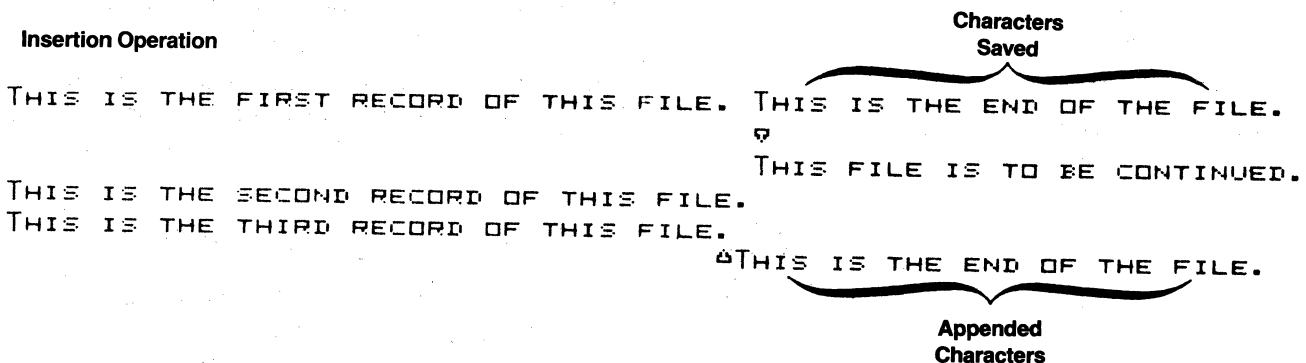
When the insertion of records is completed (the last record is inserted), the INSERT function must be terminated by activating a *record locating* function (F1-F4). As previously explained, use of a *record locating* function will cause the INSERT operation to terminate, the "close insert" symbol (␣) will print, and the characters saved in temporary storage will be appended to the last record inserted. The selected *record locating* function will then be activated.

Figure 2-9 is an example of several records inserted following the insertion of data within an existing record. Note that the characters saved in temporary storage are appended to the last inserted record.

Inserting Records Within A File. Entire records may be inserted within a file; for example, inserting entire paragraphs of text in the middle of a file is possible. Note that in this process, inserting data within an existing record is not desired; only insertion of entire records of data between two existing records in a file is of interest. The procedure is

- Use a *record locating* function (F1-F4) to locate the record immediately before the place in the text you want to insert new records.
- Once the last record before the desired insertion is located and printed and the printhead points to the end of the record, activate the INSERT function (F7). This will permit an INSERT operation to begin at the end of the record just printed. Note that since there are no characters to the right of the printhead at this point, no characters need be saved in temporary storage for later appending. If the record just printed is filled to capacity, the terminal will recognize that insertion is impossible in the full record and will automatically execute a carriage-return/line-feed and open a new record. If the record printed is **not** filled to capacity, press the ENTER key to signal the terminal that you want no more data in that record. (Note that no characters are inserted in the printed record.) The terminal then will execute a carriage-return/line-feed and insertion into a new record following the printed record may begin. Once you begin insertion of a new record, the existing record printed at the start of

a. Insertion Operation



b. Resulting Text

THIS IS THE FIRST RECORD OF THIS FILE. THIS FILE IS TO BE CONTINUED.
 THIS IS THE SECOND RECORD OF THIS FILE.
 THIS IS THE THIRD RECORD OF THIS FILE. THIS IS THE END OF THE FILE.

Figure 2-9. New Records Inserted After Insertion Within An Existing Record

- the insertion will be returned, unchanged, to bubble memory.
- Each time you complete insertion of a new record, press the ENTER key to signify the end of that record. (The SKIP key may be used to perform this function in LINE-formatted files.) The terminal then will perform a carriage-return/line-feed and insertion of another new record may begin. When pressing the ENTER key, note that characters to the right of the printhead are truncated.
 - Once you have inserted necessary records, terminate the INSERT function using one of the *record locating* function (F1-F4) keys. This will cause the newly inserted records to be stored in bubble memory, and the selected function will be activated.
 - Note that insertions in large files may take a few second after the insert operation is terminated for the terminal to allocate room in the file for the newly inserted data.
 - To insert records at the beginning of a file, first position the record pointer “above” the first existing record in the file by using the INDEX (F1) function and start inserting new records simply by using the INSERT (F7) function.
 - When inserting data and 10 empty records or less remain in the file, an audible tone will sound to signal the approaching end of the file.

**a. File Before Removal
of Filler Characters**

```
FIRST LINE.000000000
SECOND LINE.000000000
THIRD LINE.000000000
LAST LINE.000000000
```

**b. File After Removal
of Filler Characters**

```
FIRST LINE.SECOND LI
NE,THIRD LINE, LAST L
INE.0000000000000000
```

- When inserting records into a file which has no more empty records, the insert operation will automatically terminate, and the INDEX function will activate.
- When an insertion of records is attempted into a file with no empty records, an error message *** ? *** will be printed.

Termination Function

The STOP function ensures successful completion of edit mode functions.

STOP Function — F8. The STOP function should be used to exit the EDIT mode; F8 assures that any pending editing operations will be completed before exiting the EDIT mode. For example, a deletion of a large number of records may have been started but is not yet complete. The STOP function will wait for the operation to finish before returning the file being edited to memory.

Another important operation performed by the STOP function is removal of filler character from *continuous*-formatted files. As previously mentioned in the discussion of the DELETE CHARACTER and INSERT functions, filler characters are used in *continuous*-formatted files to fill out records only partially filled with data. The STOP function removes the filler characters from partially filled records and replaces them with data from subsequently entered records. Figure 2-10 shows a file with several partially filled records before and after the removal of filler characters by the STOP function. Note from the figure that once the filler characters have been removed the only record that contains filler characters is the last record in the file. Filler characters are used only during editing of a file. They are ignored in other modes of the terminal (e.g., ASR mode).

Figure 2-10. Removal of Filler Characters by the STOP Function from a Continuous-Formatted File

Editing a Record

Once a record is selected and located for editing using one of the *record locating* functions (F1-F4) or any other means, the record will be printed and the printhead will be positioned to the right of the last character in the record. The record then is available for editing.

The record is edited using the printhead as a character "pointer". Modifications can be made by positioning the printhead just below the character to be modified and executing the modification (change or deletion).

Several means are available to the operator to modify a record:

- The **CHAR** and **FIELD** keys can be used to position the printhead to the character in the record where modifications are desired. The audible tone will notify the operator when a record boundary is reached, and when the printhead is positioned 10 characters before the defined maximum record length, the audible tone will sound to notify the operator of the approaching maximum record length.
- The simplest modification is "overstriking" existing characters by positioning the printhead to the appropriate position in the record and typing the desired replacement characters. The characters entered will take the place of the characters existing in memory.
- The **DELETE CHARACTER** (F6) function can be used to delete characters within the record.
- Characters may be appended to the end of a record (if the maximum record length is not exceeded) by positioning the printhead to the right of the last character in the record and typing the characters to be appended. Attempting to enter a character beyond the defined record length will cause the audible tone to sound, and the character entered will not be accepted.
- The **INSERT** (F7) function may be used to insert characters within a record. See the description of the **INSERT** function (F7) for details.
- The **SKIP** key may be used in *continuous*-formatted files to generate an end-of-line (EOL) symbol in the record.

- The **ENTER** key may be used to truncate all characters to the right of the printhead, terminate editing of the current record, and access the record following for editing. The **SKIP** key also may be used for this purpose in *line*-formatted files. In *continuous*-formatted files if the record in which editing has been terminated is only partially filled with data, filler characters will be inserted by the terminal to right-fill the record.
 - Any *record locating* function (F1-F4) keys can be used to logically terminate editing of a record. The corrected record then will be returned to the bubble memory, and the selected function will be activated. If the printhead is not at the end of the record at the time the *record locating* function is activated, all characters remaining to the right of the printhead will print before the record is returned to bubble memory.
 - The **STOP** function (F8) also may be used in the same manner as the *record locating* functions to logically terminate editing of a record and exit the **EDIT** mode.
-

Abnormal Editor Termination

The Model 763/765 text editor protects the user from losing large amounts of inserted data caused by abnormal exits from the EDIT mode. *Abnormal EDIT mode exits* are defined as those other than exit by use of the STOP function (F8). Abnormal exits include pressing the CMD key and loss of terminal power.

The text editor prevents losses of inserted data (caused by power loss or pressing the CMD key) by updating the file catalog entry in bubble memory each time a block of five records is inserted in the file. In this manner no more than five inserted data records can be lost because of an abnormal EDIT mode exit before an INSERT operation has terminated. If the INSERT operation has been correctly terminated, no data will be lost. For example, if upon entry to the EDIT mode, the operator inserts fifty records of data and then without terminating the INSERT operation, accidentally depresses the CMD key, the contents of the last five records inserted possibly could be lost.

The text editor also will update the file catalog entry each time records are deleted from the file. Pressing the CMD key will not cause loss of data integrity in the file when deleting records, but loss of power while deleting records may interrupt the terminal before it completes the record deletion. In this case the contents of the records that were to be deleted may be distorted. Once deletion of records is completed, loss of power will not alter data in the file.

Automatic Command Execution

Automatic command execution is achieved by means of the RUN command, which permits terminal commands to be stored in a file for later execution. The commands must be stored in a *line*-formatted file. Commands which may be stored include most of the operator commands as well as the ability to execute several of the ASR functions. For details on the creation of control files and their execution, see Section V of this manual, entitled **Prompt and Run Files**.

SECTION III COMMUNICATIONS

Hardware Interface

The hardware interfaces and signals available for the various communications configurations provided by the Models 763 and 765 terminals are discussed in the following paragraphs. Figure 3-1 shows the rear of the terminals, indicating the location of the input/ output connectors **P1** and **P2**. (Note that connector P1 is protected by a flap located directly below connector P2.)

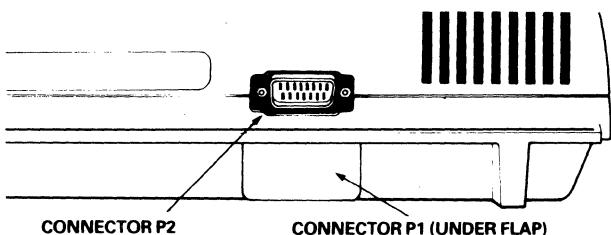


Figure 3-1. Communications Connector Locations (Rear View)

Figure 3-2 is a summary of standard factory-made cables available to connect the Models 763/765 to various data sets with which the terminal is designed to operate. Several factory-made interface cables for use with other terminals are listed in Figure 3-3; pin assignments for these cables are listed in Tables 3-1 and 3-2. Note that all required operating parameters (duplex, baud rate, etc.) must be the same between connected units.

EIA Port Interface and Signals

The EIA port of the Models 763 and 765 terminals may be used as an interface to any external device meeting the *C.C.I.T.T. V.21* or *EIA RS-232-C Interface Specification*. External devices include external modems (also called *data sets*), line printers, and other Models 763/765 terminals. Each pin in the connector is defined in Table 3-3.

The EIA interface to the terminal must be configured to meet the specified RS-232-C or V.21 requirements. Figure 3-4 shows the minimum voltage requirements and tolerances for the line receivers (SN75189A) used in the EIA interface of the Models 763/765 terminals. Each 763/765 terminal can drive no more than two other 763/765 terminals under RS-232-C specified worst-case load conditions.

EIA Interface With An External Data Set

To interface the terminal with external data sets the following EIA cables are available:

Interface — Option 113 cable assembly (TI Part No. 2200055)

Interface — Option 103/202/212 cable assembly (TI Part No. 2200051)

The optional 103/202/212 cable assembly connects the terminal to the following types of Bell System data sets: 103A, 103J, 202S, and 212A or equivalents.

The 113 cable assembly is used to connect the terminal to a Bell System Type 113A data set or equivalent. The type of data set used with the terminal will determine which cable assembly is needed. Both cables listed above connect to the terminal at P1, the bottom connector at the rear of the terminal (see Figure 3-1).

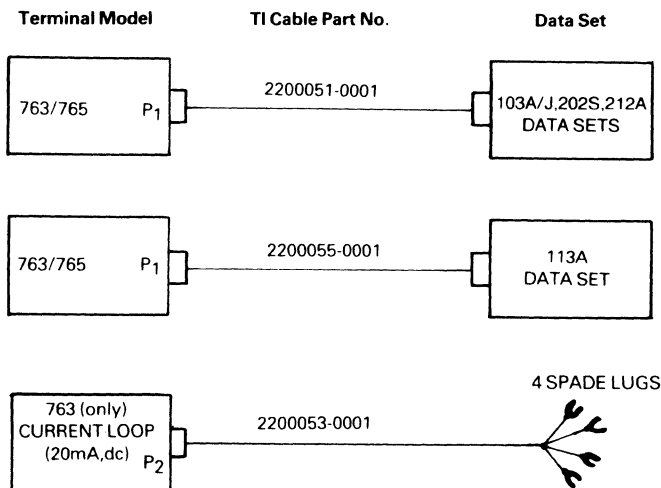


Figure 3-2. Communications Cables Available for Models 763/765

The corresponding pin assignments for each one of the optional cables are listed in Tables 3-4 and 3-5.

Table 3-6 lists the recommended data set options when operated with the Models 763/765.

Current Loop Interface for the Model 763 Terminal

To interface the Model 763 terminal to a 20-mA dc current loop use:

Current Loop — cable assembly (TI Part No 2200053).

The 20-mA dc current interface is passive; i.e., current for operation of this interface must be provided by an external source. The pin assignments corresponding to the current loop cable assembly are listed in Table 3-7.

Full and half duplex current loop connections are shown in Figure 3-5.

Acoustic Coupler

In its standard configuration the Model 765 is equipped with a built-in acoustic coupler. The only additional equipment required for communications is a standard

telephone set. The built-in acoustic coupler is compatible with the 103/113-type data sets or C.C.I.T.T. V. 21. The acoustic coupler operates only in *originate* mode. Operating frequencies are listed below.

	Mark	Space
U.S.A. transmit carrier	1270Hz	1070 Hz
U.S.A. receive carrier	2225 Hz	2025 Hz
C.C.I.T.T. transmit carrier	980 Hz	1180 Hz
C.C.I.T.T. receive carrier	1650 Hz	1850 Hz

(Text continued on page 27.)

TABLE 3-1. INTF-OPTN 810/820 CABLE PIN ASSIGNMENTS
(TI PART NO. 2263350-0001)

763/765 TERMINAL CONNECTOR (P1)	810/820 TERMINAL CONNECTOR	FUNCTION	763/765 CIRCUIT	
			EIA	C.C.I.T.T.
-1	-1	PG	AA	101
-2	-11	CTS	CB	106
-3	-2	RCV	BB	104
-4	-4 and -5	DCD	CF	109
-8	-6	DTR	CD	108.2
-9	-20 and -8	DSR/CCT	CC	107
-14	-3	XMT	BA	103
-15	-7	SG	AB	102

TABLE 3-2. INTF-OPTN TERMINAL ADAPTER CABLE PIN ASSIGNMENTS
(TI PART NO. 2263351-0001)

FUNCTION	TERMINAL EIA CONNECTOR CIRCUIT		PIN	PIN	TERMINAL EIA CONNECTOR CIRCUIT		FUNCTION
	EIA	C.C.I.T.T.			C.C.I.T.T.	EIA	
PG	AA	101	1	1	101	AA	PG
XMT	BA	103	2	3	104	BB	RCV
RCV	BB	104	3	2	103	BA	XMT
RTS/CTS	CA/CB	105/106	4 and 5	8	109	CF	DCD
DSR/CCT	CC	107	6	20	108.2	CD	DTR
SG	AB	102	7	7	102	AB	SG
DCD	CF	109	8	4 and 5	105/106	CA/CB	RTS/CTS
SCA	SCA	120	11	12	122	SCF	SCF
SCF	SCF	122	12	11	120	SCA	SCA
—	—	—	15 and 17 and 24	17 and 15	—	—	—
DTR	CD	108.2	20	6	107	CC	DSR/CCT

TABLE 3-3. EIA AND C.C.I.T.T. COMMUNICATIONS LINE INTERFACE DEFINITIONS

Conn.	Pin	Source	EIA Circuit	C.C.I.T.T.	Definition
P1	1	Common	AA	101	Protective Ground — Connected to terminal chassis and power cord ground.
P1	2	External	CB	106	CTS — Clear to Send. Switched to an ON condition by an external device to indicate that transmission is permitted. Bell System 103 data set holds this signal constantly ON once the communications channel is established.
P1	3	External	BB	104	RCV — Received data. EIA level held to a marking condition by an external device when there is no incoming data.
P1	4	External	CF	109	DCD — Data Carrier Detect. EIA level switched to an ON condition by an external device to indicate that received data is now valid.
P1	5	External	SCF	122	SCF — Reverse Channel Receive Carrier Detect. EIA level switched to an ON condition by an external device when using reverse channel to indicate the terminal is enabled to transmit data.
P1	6	Internal			Reserved
P1	7	Internal	SCA	120	SCA — Reverse Channel Request To Send. EIA level switched to an ON condition when using reverse channel to indicate the terminal is ready to receive data.
P1	8	Internal	CD	108.2	DTR — Data Terminal Ready. EIA level switched to an ON condition to prepare an associated data set for connection to a communications line; also maintains the connection once it is established. DTR is switched OFF to terminate a call.
P1	9	External	CC	107	DSR — Data Set Ready. Switched to an ON condition by an external device when connection is made to the communication line.
P1	10	External	CE	125	RI — Ring Indicator. Switched to an ON condition by an external device upon receipt of each ring pulse from the line. Must be held ON for at least 500 msec for terminal to recognize.
P1	11	Internal	CA	105	RTS — Request to Send. Switched to an ON condition when data is ready for transmission. In half-duplex operation this signal is also used by the associated data set to control the direction of transmission.
P1	12	External			Reserved
P1	13	External			Reserved
P1	14	Internal	BA	103	XMT — Transmit Data. EIA level held to a marking state when no data is being transmitted.
P1	15	Common	AB	102	Signal Ground — Common return for all data and control lines.
All following pins are reserved when the Model 765 acoustic coupler is in use.					
P2 (J403)	1	Common	AB		Signal Ground — Common return for all data and control lines.
¹ P2	4	Common	DR		DR — Data Ring. Analog data signal of internal modem.
² P2	4	Internal			X2 — TTY Transmit Current Loop gated by the internal TTY switch.
¹ P2	5	Common	DT		DT — Data Tip. Analog data signal of internal modem.
² P	5	Internal			X1 — TTY Transmit Current Loop sourcing the internal TTY switch.
P2	6	External			RL1 — TTY Receive Current Loop sourcing the internal TTY detector.
P2	7	Internal			RL2 — TTY Receive Current Loop sinking the internal TTY detector.
P2	9	Internal	AA		Protective Ground — Connected to terminal and power cord ground.
P2	2,3,8, 10-15				Reserved

¹For internal modem use.²For current loop interface use.

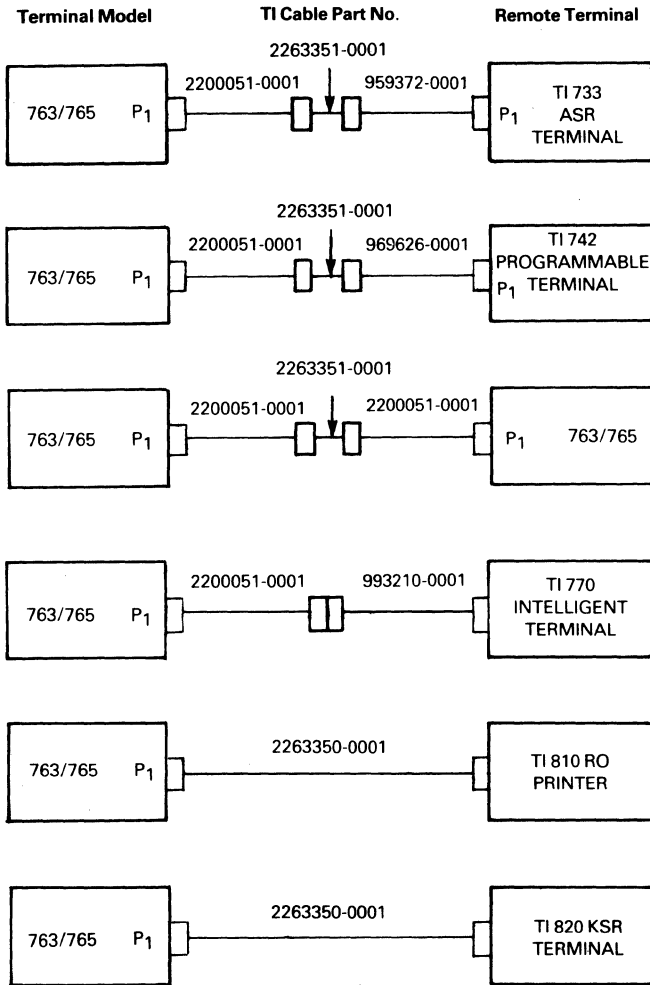


Figure 3-3. Interface Cables for Use With Other Terminal

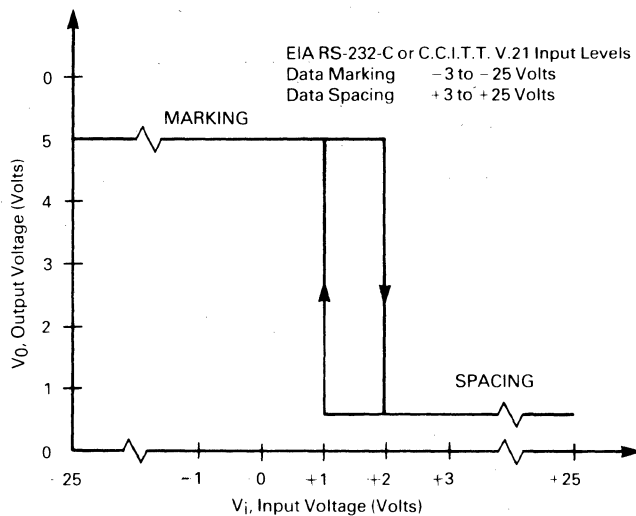


Figure 3-4. Models 763/765 Line Receiver Output and Input Voltage Levels

TABLE 3-4. INTF-OPTN 113 CABLE PIN ASSIGNMENTS (TI Part No. 2200055)

Terminal Connector (P1)	Cable Termination (113)	Function	Circuit	
			EIA	C.C.I.T.T.
-1	-1	PG	AA	101
-2	-5	CTS	CB	106
-3	-3	RCV	BB	104
-4	-13		ON	ON
-8	-20	DTR	CD	108.2
-9	-6	DSR/CCT	CC	107
-11	-4	RTS	CA	105
-14	-2	XMT	BA	103
-15	-7	SG	AB	102

TABLE 3-5. INTF-OPTN 103/202/212 CABLE PIN ASSIGNMENTS (TI Part No. 2200051)

Terminal Connector (P1)	Cable Termination (202/212)	Function	Circuit	
			EIA	C.C.I.T.T.
-1	-1	PG	AA	101
-2	-5	CTS	CB	106
-3	-3	RCV	BB	104
-4	-8	DCD	CF	109
-5	-12	SCF	SCF	122
-7	-11	SCA	SCA	120
-8	-20	DTR	CD	108.2
-9	-6	DSR/CCT	CC	107
-10	-22	RI	CE	125
-11	-4	RTS	CA	105
-14	-2	XMT	BA	103
-15	-7	SG	AB	102

The transmit level is adjustable from -20 dBm to 0 dBm **ONLY ON U.S.A. MODELS**. Receiver sensitivity is -38 dBm in full-duplex 300-baud operation and -45 dBm in half-duplex 300-baud operation.

The receiver converts the analog *frequency shift keyed* (FSK) input signal to digital EIA-level serial-data output at data rates up to 310 baud. The digital receive data output signal is held in a *marking* condition until the data carrier detect output signal is on. The transmitter converts the EIA-level serial-data output from the terminal control electronics to an analog phase-coherent FSK signal at data rates up to 310 baud. No signal is transmitted until the data carrier detect delay output signal is on.

(Text continued on page 29.)

TABLE 3-6. RECOMMENDED DATA SET OPTIONS

103J	
Modem Option	Recommended Setting
Receive space disconnect	Either
Send space disconnect	Either
Loss of carrier disconnect	Either
CC Indication	Early
CB and CF indications	Common
CC indication for analog loop	On
Automatic answer	Yes
Failsafe state of CN circuit	Off
Common ringer	Either
Common grounds	Yes
Tip ring make busy	No
202S	
Soft turnoff and squelch intervals	Soft turnoff = 24 ms Squelch = 156 ms
Fast carrier detect	Out
Clear-to-send interval	180 ms
Auto answer	In
Local copy primary channel	Out
Condition of CC (DSR) in analog loop	On
Transmit only	Out
Echo suppression enable	Out
Carrier control turnaround	In
Early CC (DSR) indication	Out
Reverse channel	Either
Local copy on reverse channel	Out
Grounding option	Signal ground to frame
212A	
Tip ring make busy	Out
CC indication for analog loop	On
CN circuit	Out
Transmitter timing	Internal
1200 baud operation	Async/start-stop
Character length	10-bit
Receiver respond to digital loop	Off
Loss of carrier disconnect	In
Receive space disconnect	In
CB and CF indications	Common
Send space disconnect	In
Automatic answer	In
Answer mode indication, CE	Off
Speed mode	Dual
Interface speed indication, CI	In
Signal ground to frame connection	In

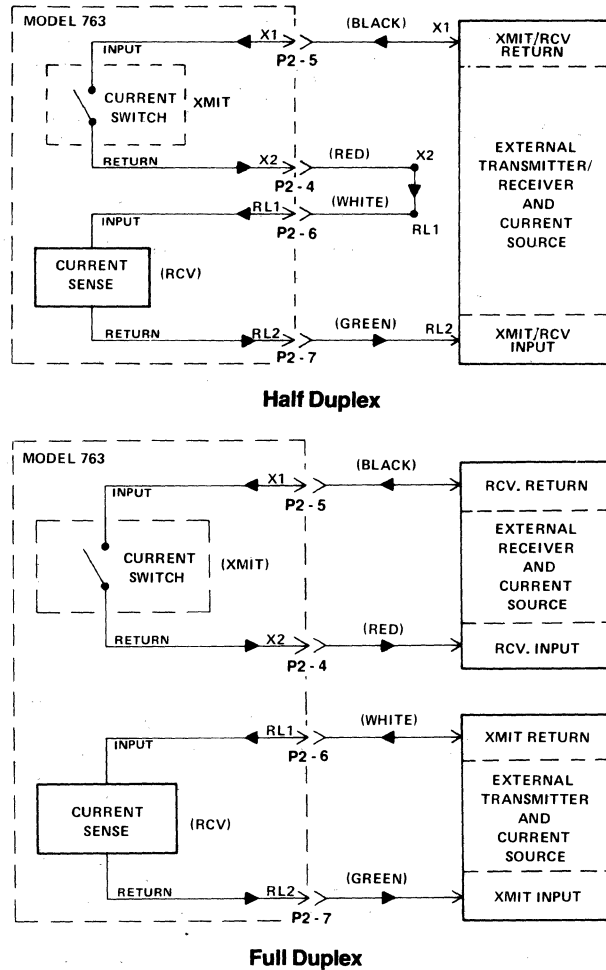


Figure 3-5. Model 763 Full-Duplex and Half-Duplex Current Loop Connections

TABLE 3-7. CURRENT LOOP CABLE PIN ASSIGNMENTS (TI PART NO. 2200053)

Terminal Connector (P2)	Cable Termination	Function	Lead Color
-6	E1	RL-1	White
-7	E2	RL-2	Green
-5	E3	X-1	Black
-4	E4	X-2	Red

NOTES

- dc Current Loop Signal Levels and Tolerances:
 - Maximum current: 60 mA (transmit or receive)
 - Nominal current: 20 mA
 - Maximum voltage drop: 3 volts (receive) and 1.5 volts (transmit) while marking
 - Maximum transmit voltage: 50 volts while spacing

Carrier Detect Delay

The Models 763/765 electronics provide two carrier detect delays. The normal (standard) delay is the long delay; a short delay is optional. Table 3-9 lists the delays provided.

TABLE 3-9. DATA CARRIER DETECT SIGNAL DELAYS

Standard Delay	OFF → ON 2.5 to 6.0 seconds ON ← OFF 100 to 300 milliseconds
Optional Delay	OFF → ON 0.5 to 2.0 seconds ON ← OFF 30 to 100 milliseconds

EIA Line Discipline and Timing

The Models 763/765 provide three modes of line operation selectable from the keyboard using the CHANGE command: Full duplex, half duplex, and half duplex with reverse channel. Several characteristics are shared among the three operating modes:

RTS — Three distinct modes of operation are possible with the EIA *request-to-send* signal; character transmission, line transmission (buffered operation), and file transmission. RTS is on only when the terminal is transmitting; it is off at all other times. For single character transmission (normal keyboard transmission, BUFFER option OFF), RTS is activated when a key is depressed, the digital code for the key is transmitted by the terminal, and then RTS is switched off. In the line transmission mode (BUFFER option ON), an entire line of data is entered into the data terminal buffer (see **Buffer Option Operation**). When the SKIP key is pressed, or the buffer overflows, RTS is activated, the entire line is transmitted, and RTS is switched off with no delay. In the file transmission mode RTS is activated, PLAYBACK is activated for transmission, and RTS is switched off when PLAYBACK terminates. The terminal automatically selects the *character* and *line* modes (depending on the state of the buffer option) and the file mode.

CTS — The terminal cannot transmit characters unless clear-to-send is present. CTS is checked before each character is transmitted. If CTS is on, the character is transmitted; if CTS is off, the terminal will wait up to 10 seconds for CTS to come on. When CTS does come on, the character will be transmitted. If CTS does not come on within the allotted 10 seconds, the terminal will dis-

connect the line. If CTS goes off while a character is being transmitted, that character *will* be transmitted. The CTS signal can be used to clock data from the terminal by switching off CTS when data is not wanted and switching on CTS when a character is desired, as long as care is exercised never to leave CTS off for more than 10 seconds when a character transmission is being attempted.

DCD — The *data carrier detect* signal must be on for the terminal to receive data via the EIA interface.

DSR — The *data set ready* signal must be on for the terminal to transmit any data through the EIA interface.

DTR — The terminal activates the *data terminal ready* signal when it is on-line.

RI — The *ring indicator* signal is the indication to the terminal that the external modem has answered the telephone. RI is sampled by the terminal every 500 milliseconds; therefore, the signal from the modem must be at least 500 milliseconds long to ensure satisfactory operation. When the terminal detects RI, it will transmit the ABM message if the AUTOABM option is ON (see **AUTOABM Trigger**), and the terminal will initiate operation of the activity timer (see **Line Disconnect**). Once the terminal detects the RI signal, RI will be ignored until the terminal disconnects from the line; then the 500-millisecond sampling begins again. A 1.5 second delay occurs between the detection of DSR and the printing of the ABM message.

Full Duplex

In full-duplex operation, the transmitter and receiver sections of the terminal operate independently of each other; characters can be transmitted and received simultaneously. RTS is switched off after the stop bit of the last character transmitted is sent. *Reception* of characters is always enabled. The keyboard and the playback unit are linked with the transmitter and the record unit and the printer are linked with the receiver; thus, characters being *transmitted* cannot be printed or recorded.

Half Duplex

In half-duplex operation the terminal can either receive or transmit but not both simultaneously. Transmission assumes priority, but the terminal can receive characters until transmission occurs, at which time the receiver is inhibited until transmission is completed. The keyboard, printer, playback unit and record unit are all linked together in half duplex. Characters transmitted from either the keyboard or the playback unit can be printed or recorded, depending on the state of the terminal and its options.

At speeds below 1200 baud RTS is switched off after the stop bit of the last character is transmitted; the receiver is enabled 4 milliseconds later. At 1200 baud and above, an additional requirement exists for transmission: the DCD signal must be off before the terminal can transmit. DCD is checked before each character is transmitted. If DCD is on, the terminal will wait 12 milliseconds for DCD to go off. If DCD goes off within 12 milliseconds, the character will be transmitted; if DCD does not go off, the transmission attempt will be aborted. If the character came from the keyboard, an error will be reported (** 08 **). If the character came from the playback unit, playback will be switched off. RTS will go off 20 milliseconds after the stop bit of the last character is transmitted, and the receiver will be enabled 4 milliseconds later.

Half Duplex with Reverse Channel

In half-duplex-with-reverse-channel operation, the terminal can either receive or transmit as in standard half duplex. The keyboard, printer, playback unit, and record unit are all linked together as in half duplex. And transmission has priority over reception. However, the signals *reverse channel receive* (RCR) and *reverse channel transmit* (RCT) are used as ready/busy indicators. RCR and RCT correspond to the EIA circuits SCF and SCA, respectively.

When transmission is attempted, the receiver is inhibited, RTS switched on, and RCR is checked. RCR must be present to transmit; it is checked before each char-

acter is transmitted. If RCR is off, the terminal will wait either 350 milliseconds (for character transmission, the first character of a buffered line transmission, or the first character of a file transmission) or 50 milliseconds (for subsequent characters of a buffered line or file). If RCR is recovered, transmission occurs; if RCR is not recovered, transmission is aborted. If the character came from the keyboard, an error will be printed (** 08 **). If the character came from the playback unit, playback will be switched off.

The RCT signal is generated by the terminal to permit the remote data set to transmit data to the terminal. RCT is switched off whenever *request to send* (RTS) is switched on and is switched back on whenever RTS is switched off. The RTS discipline is the same as that for half duplex operation. RCT is also switched off when the receive buffer accumulates 60 characters, allowing a 20 character leeway in inhibiting received data. RCT will come back on when the receive buffer is totally emptied.

The half duplex with reverse channel mode enables terminal communications at speeds above 2400 baud, permitting reception of blocks of 2400-baud (and above) data in increments which the terminal can process. The ready/busy signals (RCR and RCT) inform the transmitting device when the terminal is *busy* and when the terminal is *ready* to receive a new block of data.

AutoABM Trigger

The autoABM trigger is a parameter enabled by the CHANGE AUTOABM command. When AUTOABM is enabled, the terminal scans the ring indicator (RI) line (as explained above under **EIA Line Disciplines**). When RI is detected, the terminal initiates a 22-second timeout, waiting for DSR to come on. If DSR does not come on, the terminal will attempt to disconnect the line. If DSR comes on, the terminal will initiate a 1.5-second delay. After the 1.5-second delay the ABM message (previously created using the CHANGE command) will be transmitted, and if conditions permit (half duplex, ABMPRT on, unsecured ABM, printer on), the ABM also will be printed. The ABM cannot be recorded in memory.

ENQ Timing

When the ENQ control character (enquiry) is received from the line, the ABM message will be transmitted, and if conditions permit (see above), the ABM message will be printed. Assuming that the ENQ character is the only character in the receive buffer, a minimum 34-millisecond delay will occur from the reception of the character to the transmission of the first character of the ABM message. A longer delay will occur if characters are

waiting in the receive buffer for processing or if the terminal is doing some other task.

Line Disconnect

There are four conditions under which the terminal will disconnect itself from the line:

- If CTS is off for more than 10 seconds during an attempted transmission (see **EIA Line Discipline**)
- If DSR is lost, or if the carrier is lost from the acoustic coupler
- If the EOT control character is received and the EOTDIS option is enabled
- If the activity timer expires. As previously mentioned the activity timer is enabled when the terminal detects RI, indicating that an external modem has answered the phone. The activity timer is a 5-minute timer which is reset to 5 minutes every time a character is received or transmitted, or when the interactive ASR terminal mode is entered (after a command is executed). If 5 minutes pass with no terminal activity, the terminal will disconnect the line. The activity timer is disabled when a line disconnect is performed.

The actual line disconnect causes the following actions:

- Playback and record are switched off
- The printer is enabled
- The keyboard is unlocked
- The activity timer is switched off
- The DTR signal is brought logic low for 1 second to signal the external data set to "hang up" the phone. DTR then returns to logic HIGH to indicate that the terminal is ready for the next call.

Buffer Option Operation

When the terminal is on line and the BUFFER option is ON, the terminal is in the buffered mode. This mode causes a line of data to be entered into a buffer for editing before the line is transmitted. When a keyboard key is depressed, that character is put into an 80-character buffer and if the printer is on (ASR FCTN 9), the character is printed (regardless of the DUPLEX

selection or whether communications can occur). Characters are accumulated in the buffer until either the SKIP key is pressed or the 81st character is typed, causing the buffer to overflow. When the SKIP key is pressed, the buffer contents plus the transmit-EOL symbol are transmitted in the line transmission mode. If an 81st character is typed, the previous 80 characters plus the EOL are transmitted in the line transmission mode. The 81st character will be printed following a CR/LF and will become the first character in the buffer. If the 81st entry is the SKIP key, the second buffer-full contents (containing only the transmit EOL) will be transmitted following transmission of the first buffer; subsequent characters will be put in the next buffer.

Before transmission the buffer can be edited using the **CHAR**, **CHAR**, **FIELD**, and **FIELD** keys. The **CHAR** key will backspace the printhead one space and perform a line feed (if necessary) to position the printhead under a printed character; the subsequent key entry will print the new character below the old one, and the new character will replace the old character in the buffer. Subsequent **CHAR** key entries will cause further backspaces. If the **CHAR** key is pressed and held, the printhead will backspace at a rate of 15 spaces per second. If the **CHAR** key is pressed when the buffer pointer is at the beginning of the buffer or if the buffer is empty, the audible tone will sound. The **CHAR** key (SHIFTed CHAR) will move the printhead forward one space and cause the next character in the buffer to be printed. If the pointer is at the end of the buffer (the last character entered in the buffer), the audible tone will sound.

The **FIELD** key will cause a CR/LF, and the pointer will be positioned at the beginning of the buffer; this is equivalent to numerous **CHAR** (backspacing) entries to the beginning of the buffer. If the pointer is already at the beginning of the buffer or the buffer is empty when the **FIELD** key is pressed, the audible tone will sound. The **FIELD** key (SHIFTed FIELD) will cause the remainder of the buffer to be printed and the pointer to be positioned after the last character in the buffer. If the pointer is already at the end of the buffer, the audible tone will sound.

If transmission cannot proceed for some reason, an error will be reported when the line transmission is attempted. If RECORD is ON (ASR FCTN 2), and the terminal is in half-duplex, the line will be recorded after the SKIP key is pressed or the buffer overflows, but before the actual transmission occurs. This permits the edited line to be recorded. If PLAYBACK is ON (ASR FCTN 1), the buffered mode is disabled until playback terminates.

Host System Communications with the Models 763/765 Terminal

Employing the various *extended device control* (EDC) functions a remote host system can instruct the terminal to perform many of the operations manually exercised from the terminal keyboard. EDC is a combination of ESC sequences which have certain meanings to the terminal plus the DC1, DC2, DC3, and DC4 control characters, which are also known as *automatic device control* characters. However, the systems programmer must be cognizant of the throughput limitations of the terminal. When operating solely as a keyboard send-receive (KSR) terminal, the Models 763/765 can process constant data streams at 300 baud; but when commands are added, throughput limitations may be exceeded. The terminal has an 80-character receive buffer into which all received characters are channeled. When a command is recognized, the terminal will execute that command. Execution time may range from milliseconds to many seconds, or even minutes, depending on the command given. During command processing the receive buffer can easily overflow and lose characters.

This section explains how to give the terminal commands from the communications line and outlines timing considerations for EDC commands. Section IV discusses timing considerations for each of the operator communications commands (those commands executed in the COMMAND mode). All command execution times are measured from the time the command is recognized (the commands may have been in the receive buffer for a time before being recognized). Another consideration involves printing of data interspersed with commands: Although the printer is faster than printable characters received at 300 baud, a carriage return can consume up to 180 milliseconds, causing the receive buffer to fill.

EDC Commands

The extended device control (EDC) commands enable a remote host system to give many of the available commands to the terminal over the communications line. The EDC commands include the standard ASCII control characters DC1, DC2, DC3, and DC4 plus the special ESC sequences which consist of the ESC (escape) control character followed by an additional character specifying the command. The EDC option must be set to ON (via local CHANGE command) for the command to be processed.

The individual option control (DC1.3 or DC2.4) must be set to ON via the CHANGE commands to enable the use of these standard device control characters. The following EDC commands are recognized by the 763/765.

DC1 — When the DC1 control character is received, playback will switch on and the error status will be set to 00 (see **ESC 1** command). If no playback file is set up, the error status will be set to error 88. The DC1.3 option also must be set to ON (via the CHANGE command) for this character to have effect. From 14 to 20 milliseconds is required for playback to come on and the first character to be transmitted.

DC2 — When the DC2 control character is received, record will switch on and the error status will be set to 00. If no record file is set up, the error status will be set to error 88. If the record file is full (the record pointer is at the end of the file), the error status will be set to error 77. No additional delay is incurred to switch record on. The DC2.4 option must be set to ON (via the CHANGE command) for this character to have effect.

DC3 — When the DC3 control character is received, playback will switch off and the error status will be set to 00. No additional delay is incurred to switch playback off. The DC1.3 option must be set to ON (via the CHANGE command) for this character to have effect.

DC4 — When the DC4 control character is received, record will switch off and the error status will be set to 00. The delay necessary to switch record off is a function of the record file size. For a file of 80 characters per record, up to 250 milliseconds may elapse from the instant the DC4 character is recognized. The DC2.4 option must be on (via the CHANGE command) for this character to have effect.

NOTE

When transmitting ESC sequences to the Models 763/765, do not use a space between the ESC and the subsequent character. For example, the sequence should be:

```
ESCCHANGE EDC TO ON
```

ESC0 — This sequence instructs the terminal to accept characters received following the ESC0 (and terminated by the receive-EOL) as a local operator command and to execute that command. For example, if RCVEOL is set to CRLF, the sequence

```
ESCCHANGE ABM TO "THIS IS A NEW ABM" ESC
```

will change the ABM message to **THIS IS A NEW ABM**. (See Section IV for operator command timing considerations.)

NOTE

The following operator commands are not accepted from the communications line and will cause the error status to be set to FF: EDIT, CATALOG, STATUS, TEST, ON-LINE, and OFFLINE.

ESC 1 — The ESC 1 command is a request for terminal status. The terminal will respond by transmitting six characters representing the state of the terminal, followed by the transmit EOL (XMTEOL). The first two characters represent the error code of the last operator command or EDC command to be processed. An error code **00** means no error occurred. An error code **FF** indicates that the command was not a valid command. Refer to the table of error codes in the *Models 763/765 Operating Instructions* for other code definitions. The next four characters represent the states of various parameters of the terminal. These characters are interpreted by their hexadecimal representations; the individual bits have the following meanings:

Third Character

- (MSB) Bit 1 — Playback is assigned to an existing file.
 Bit 2 — Record is assigned to an existing file.
 Bit 3 — The playback and record files are the same file.
 (LSB) Bit 4 — The playback file is a line format file (a logic zero indicates a continuous-format file).

Fourth Character

- (MSB) Bit 1 — The record file is a line file (a logic zero indicates a continuous file).
 Bit 2 — The record file is LOCKED.
 Bit 3 — The printer has been switched off by a PRINTER-OFF command.
 (LSB) Bit 4 — DC2.4 is on.

Fifth Character

- (MSB) Bit 1 — DC1.3 is on.
 Bit 2 — DC3 is on.
 Bit 3 — EDC is on.
 (LSB) Bit 4 — Buffer is on.

Sixth Character

- (MSB) Bit 1 — EOTDIS is on.
 Bit 2 — AUTOABM is on.
 Bit 3 — ABMPRT is on.
 (LSB) Bit 4 — PCHECK is on.

The time required for the terminal to process the ESC 1 command and begin transmitting the status ranges from 6 to 8 milliseconds.

ESC 2 — This command is the playback-forward-one-record command. ESC2 performs the same processing as the FCTN 7 key. Playback will activate as if a DC1

control character were received; i.e., a delay of 14 to 20 milliseconds will occur before the first character is transmitted.

ESC 3 — This command is the playback-reverse-one-record command. ESC 3 performs the same processing as the FCTN 8 key. Up to 120 milliseconds may elapse for ESC 3 processing, depending on record size of the playback file and if the playback pointer is at the beginning of a record.

ESC 4 — The ESC 4 command is the rewind-playback command. It performs the same processing as the FCTN 5 key. 120 milliseconds are required to rewind playback.

ESC 5 — The ESC 5 command is the rewind-record command. It performs the same processing as the FCTN 6 key. No extra delay is needed to rewind record.

ESC 6 — The ESC 6 command activates the ADC. When received the terminal activates options DC3, DC1.3, and DC2.4, enabling all ADC functions. ESC 6 consumes up to 50 milliseconds for execution.

ESC 7 — This command is the ADC-off command. When received ESC 7 causes the terminal to deactivate DC3, DC1.3, and DC2.4, disabling all ADC function. ESC 7 consumes up to 50 milliseconds for execution.

ESC 8 — The ESC 8 command is the printer-on command. When received the terminal printer will switch on. ESC 8 requires no extra delay to execute.

ESC 9 — The ESC 9 command is the printer-off command. When received the terminal printer will switch off. ESC 9 requires no extra delay to execute.

ESC : — This command totally locks out the keyboard from terminal operation, yielding total terminal control to the communications line. ESC : requires no extra delay to execute. The CMD is also locked.

ESC ; — ESC ; frees the keyboard. This command requires no extra delay to execute.

ESC < — This command will cause return to the RUN file if a RUN file is active. Timing considerations must be addressed by the systems programmer when creating the RUN file.

ESC ESC — This sequence is necessary to record ESC characters into a file when EDC is enabled.

SECTION IV

THROUGHPUT AND TIMING

Operator Communications Command Execution Timing

The time required by the terminal to execute an operator communications command comprises three distinct time periods: (1) Time to terminate the ASR mode; (2) Time to execute the operator command; and (3) Time to re-enter the ASR mode.

The length of the individual time periods is dependent on a number of system variables such as the number of

existing files and their location in the file catalog. For simplicity, the details upon which the calculation of the time periods is based are omitted from this discussion. Instead, Table 4-1 provides the typical total times required by the Models 763/765 to execute the various commands.

TABLE 4-1. OPERATOR COMMAND EXECUTION TIMING

COMMAND	EXECUTION TIME
CREATE	The terminal must search the catalog for an empty entry. If the catalog is empty, the command will consume 0.26 second. If the catalog has 15 entries, the time will be 1.46 seconds.
DELETE ERASE LOCK FREE	The terminal must search the catalog for the file by name. If the file is the first, 0.52 second will elapse to perform the command. If the file is the last file in the catalog, 1 second will elapse to perform the command.
COPY	<ul style="list-style-type: none"> • To copy file to file, both files must be located; the copy rate is: 0.250 second/record for record length 73-80 characters 0.220 second/record for record length 55-72 characters 0.180 second/record for record length 37-54 characters 0.150 second/record for record length 19-36 characters 0.110 second/record for record length 01-18 characters • To copy file to printer, the printer rate is approximately 30 characters per second. • Copy keyboard to file depends on operator skill.
CHANGE	A change command executes in 0.52 second for all parameters except the ABM; to change the ABM message requires 0.75 second.
TEST	The self-test executes in approximately 5 seconds, but cannot be executed directly from the line.
ONLINE	0.5 second, but cannot be done directly from the line.
OFFLINE	0.5 second, but cannot be done directly from the line.
STATUS	8-10 seconds, but cannot be done directly from the line.
CATALOG	5-15 seconds, but cannot be done directly from the line.
EDIT	Operator dependent, and cannot be done directly from the line.
RUN	Sum all individual commands, add printing of comments at 30 CPS, and then add approximately 35% of that total for overhead.

Command Execution from the Communications Line

As previously mentioned, when giving commands to the terminal over the communications line, sufficient time must be allotted for the terminal to execute the commands. Otherwise, those characters which overflow the terminal receive buffer may be lost. One of the following four methods can be employed to prevent loss of data.

1. After the command is sent to the terminal (ESC 0 + command), send an ESC 1 sequence (status request) to the terminal. The terminal then will process the command and when processing is completed, the terminal will process and transmit the terminal status (see **ESC1** sequence in Section 3). Using this method, the remote host system is actually waiting for an acknowledgment from the terminal that the command processing is completed. An added advantage is that the host can determine if the command was processed successfully or if an error occurred.
2. After the command is sent to the terminal, the host system can invoke a time delay to await terminal command processing time. This delay should equal the command completion time estimated from the information in this subsection and in Table 4-1.
3. After the command is transmitted to the terminal, the host system can send filler characters for a period equal to the estimated command completion times. The filler character should be a nonprintable (control) character which have no significance to the Models 763/765 (for example, NUL or DEL). With this scheme the buffer will overflow and characters will be lost, but lost characters will be only the "throwaway" filler characters.
4. All commands to be executed can be recorded in a file in the format for a RUN file; then the RUN file can be executed by the command sequence **ESCORUN (filename)** (see Section V under **RUN Files**). This method eliminates problems with execution time of individual commands; however, one must be careful to allot time for execution of the entire RUN file before the host sends more characters or commands. Any of the first three methods listed above can be used to estimate times, but particular

care must be taken using the ESC 1 method: if the status request is in the receive buffer and the RUN file temporarily switches to the ASR mode, the status request may be processed at that time, falsely indicating that the RUN file was completed. An alternative is to include as the last line of the RUN file:

```
#ESC CHARACTER STRING *
```

This transfers the characters **CHARACTER STRING** to the ASR transmit buffer (without transmitting immediately) and will return control to the RUN file. Since it is the last statement in the RUN file, the RUN file will terminate, the interactive ASR mode will be entered, and the transmit buffer will be processed and transmitted. In this way, the string **CHARACTER STRING** serves to acknowledge to the host system that the RUN file has terminated.

Recording Data from the Communications Line

Using the appropriate commands it is possible to set up a record file to store data from the communications line. The following steps outline a general procedure for transferring data from the host system to the terminal. This is certainly not the only procedure, nor even a recommended procedure, but merely an example:

1. Create a file named DATA by sending the sequence:

```
ESC CREATE DATA L 50 724 ESC 1
```

In this example the terminal previously must have been set up with EDC to ON (essential to begin any remote control) and RCVEOL to NL. The terminal will send a status message to acknowledge that the command has been executed; the host must wait for this acknowledgement.

2. Assign DATA file to be the RECORD file with the sequence:

```
ESC CHANGE RECORD TO DATA4 ESC 1
```

* See page 48 for an explanation of the #ESC function.

Again, the host must wait for an acknowledgement.

3. REWIND the RECORD file, switch on the RECORD file, and switch off the ADC functions with the sequence:

```
ESC6ESC7
```

Switching off the ADC functions makes it possible to record the ADC control characters DC1, DC2, DC3, and DC4 without affecting the RECORD and PLAYBACK files. This is necessary if the file being downloaded is a prompt file containing these four ASR control characters. The ESC 7 sequence will be interpreted by the terminal as a command sequence, so it will not be recorded even though RECORD is ON.

4. The host system will now invoke a 100-millisecond delay to ensure that the ADC OFF command was processed. This delay equals only three characters at 300 baud and may not be necessary. But if the data stream may possibly tax the throughput of the terminal, either by inefficient record lengths (see **Throughput Limitations** below) or very short records, the delay may be wise. Note that at speeds above 300 baud, the terminal printer is automatically disabled when recording from the communications line.
5. Download to the terminal the data to be recorded.
6. Switch ADC ON and switch RECORD OFF with the sequence:

ESC6DC4

The terminal will interpret ESC 6 as a command and consequently will not record it.

Since this sequence is less than 80 characters, it could be sent at one time without regard to processing times for the commands. This string would assign the file named DATA to be the PLAYBACK file, rewind the file, and switch playback ON.

Throughput Capabilities

To help understand throughput limitations of the Models 763/765 data terminals, a review of the structure of the bubble memory file system may be useful. When the terminal is instructed by the CREATE command to create a file, the terminal file management system allocates available user memory by dividing the number of characters in each record by 18 (the number of characters per memory page) and rounding up to the next integer (for example, 80 characters in the CREATE MYFILE L 4 80 command). The file management system then multiplies the number of pages needed per record by the number of records requested by the CREATE command (5 pages per record \times 4 records = 20 pages).

In the example above MYFILE occupies 20 pages of bubble memory; thus the total amount of bubble memory storage allocated by file management is 360 bytes (20 pages \times 18 bytes), but the user has only 320 bytes available (80 characters-per record \times 4 records).

When creating files careful selection of the number of characters per record is important because the terminal file management system allocates records by memory page boundaries. The most efficient use of memory is obtained by assigning record lengths that are multiples of 18 (18, 36, 54, 72) bytes. The following discussion of terminal throughput assumes that no EDC or terminal commands are included in the data stream. (Please refer to Table 4-1 for command timing.)

The maximum data rate possible with the bubble memory while recording data from the communications line (with the playback unit OFF and no EDC control characters in the data stream) is 320 characters per second. When recording data at speeds above 2400 baud, the half-duplex-with-reverse-channel mode of operation described in Section III will yield the best results. When recording data at 2400 baud, memory file structure becomes important. Consider a file set up by the CREATE command with 100 records, each record 45 characters in length. A total of 4500 characters would fill this file; and at 240 characters per second, the file would be filled in 18.75 seconds [4500 characters \div (240 characters per second) = 18.75]. For each 45-character record, three memory pages (45 \div 18 = 2.5, rounded to 3) must be recorded by the bubble memory. This process is

Acquiring Data from a Playback File

To upload data from the terminal to the host system, simply assign the playback file to the desired data file, rewind playback and switch on playback. For example, to upload the file recorded in the six steps above, the sequence sent to the terminal by the host could be:

```
ESCCHANGE PLAYBACK TO DATA+ESC4
```

equivalent to recording 54 characters per record, of which 9 characters per record are not being used. The bubble memory, in effect, must record 5400 (54×100) characters in 18.75 seconds (the time the communication channel consumes sending 4500 characters at 2400 baud). A recording rate of 288 characters per second results ($5400 \text{ characters} \div 18.75 \text{ seconds}$). Since the 288-characters-per-second rate does not exceed the 320-characters-per-second limit, no data will be lost when recording into the file at 2400 baud.

At 2400 baud the limiting file structure is a *continuous*-formatted file of 27 characters per record. Consider, for example, a file of 100 records of 27 characters per record. A total of 11.25 seconds ($27 \times 100 \div 240$) will elapse to fill the file. The bubble memory must record 36 characters ($27 \div 18 = 1.5$) for each 27 characters received. This is an effective recording rate of 36×100 characters in 11.25 seconds or 320 characters per second which is the maximum recording rate of the bubble memory. If the record length is less than 27 characters, at 2400 baud the recording rate would exceed 320 characters per second.

The same throughput limitations apply to *line*-formatted files. When recording into *line*-formatted files at 2400 baud, a long series of short lines would cause the bubble memory to record at a higher rate than the 320-characters-per-second limit.

In either *line*- or *continuous*-formatted files throughput is a function of the number of characters recorded per record. If both record and playback are on at the same time, throughput is half the calculated value of the following sheet. These calculations assume recording data without printing and without acting on control characters or commands.

Table 4-2 lists the minimum record length recommended for various baud rates if no data is to be lost, with either PLAYBACK-ON or RECORD-ON (but not both).

When the playback and record units are both ON, the effective recording rate of each is reduced by *more* than half the rate if only one unit is ON. The main reason for this fact is that the terminal employs *one* bubble memory controller. All read and write operations pass through the controller. The playback and record firmware must wait for the controller to be free before proceeding. The terminal cannot support simultaneous activity of the playback and record functions at 2400 baud. The record function may lose data if the 80-character buffer capacity is exceeded.

Receive Buffer Busy

In full-duplex mode the terminal has no way to provide a busy signal when the receive buffer is full. In half-duplex-reverse-channel mode pin 7 of connector P1 is used as a busy LOW line. The conditions that cause the data terminal to go busy [pin 7 (SCA)LOW] are when the terminal is transmitting or when the 60th character is received into the 80-character receive buffer. After all characters are removed from the receive buffer, pin 7 returns to a HIGH state indicating a "nonbusy" condition. The terminal monitors pin 5 (SCF) on P1 to determine whether the line is free or a busy condition does not exist from another device.

TABLE 4-2. MINIMUM RECORD LENGTHS AT VARIOUS BAUD RATES

BAUD RATE	MINIMUM CHARACTERS PER RECORD
2400	27
1200	7
600	4
300	2
200	2
150	1
110	1

SECTION V

PROMPTING AND RUN FILES

An important feature of the Models 763/765 terminals, particularly useful for orderly data acquisition, is its capability to accept operator prompting files. A prompt file may be created in either *line* or *continuous* format. A prompt file is really nothing more than a file which, when used for prompting, is designated the PLAYBACK file. A prompt file typically contains groups of character strings, acting as operator prompts, interspersed with the ASR control characters DC1, DC2, DC3, and DC4 along with other special codes for controlling input, printing, and recording of data supplied either by the operator or by the prompt file itself. The control characters, considered as a group, are referred to as the *automatic device control* (ADC).

Prompting files (especially when used in conjunction with RUN files) permit the terminal to be operated by persons having little or no knowledge of telecommunications, terminals, or programming. Prompting files are

usually designed so that the operator may enter all required data in response to prompts expressed in easily understood language. Prompt files are constructed so that immediately after the terminal prints an operator prompt or other request for data, the RECORD-ON function is activated to record the operator's answer to the prompt. Data acquisition sessions conducted under control of a prompt file generally assume the form of a "question and answer" or a "fill in the blanks" session.

A RUN file consists of specific terminal setup commands stored in memory. The RUN file is initiated by the terminal operator simply by entering the **RUN (filename)** command from the keyboard. A RUN file is particularly useful in prompting operations to reduce the necessity for operator manipulation of terminal configurations to accomplish particular functions. Further details are provided later in this section.

ASR Off-Line Operation

Before delving into the details of prompt and RUN file creation and usage, a review of off-line ASR operations may be helpful. Although the *Models 763/765 Operating Instructions* provide procedures for using the ASR mode functions keys, special keys, and ADC characters, the following subsections emphasize their value to prompt and RUN files.

The Function Keys

The function keys on the Models 763/765 are provided to enable the operator to perform ASR functions whether the terminal is on line or off line. The ASR functions (also discussed in the *Models 763/765 Operating Instructions*) permit manipulation of the PLAYBACK and RECORD files. ASR functions include playback and record ON/OFF, playback and record "rewind" playback forward and reverse, and printer ON/OFF.

To initiate these functions simply press and hold the keyboard FCTN key and then press the appropriate unshifted number key. In the following discussion the function keys are denoted by the uppercase letter "F" and a single digit (e.g., F5). The ASR functions are listed on the label above the keyboard number keys.

Playback On — F1. The *playback-on* function causes reading of the file previously designated as the PLAYBACK file. If the terminal is ON LINE, data played back will be transmitted via the active communications port. If the terminal printer is ON (ASR Function F9), data will be printed as it is played back from memory. But it should be noted that if the terminal is on line and in the full-duplex mode, data will not be output to the printer regardless of the state of the printer on/off functions (F9/F0).

Data playback will be terminated by one of the following conditions:

- The last record of the playback file is processed.
- The operator inputs an F3 (playback off) function, or
- A playback-off control character (DC3) is encountered in the playback file and the DC3 and EDC options are enabled.

Once playback halts, it may be resumed by another F1 unless the halt resulted from reaching the end of the file. If the end of file was reached, an F1 will not be successful until the playback file is rewound (F5) or playback reversed (F8). (See also **Playback Forward — F7** for a similar function.)

When the F1 (playback-on) function is active, the PLAYBACK indicator lamp will remain lit.

Record On — F2. The *record-on* function causes all data received from either the keyboard or the communications line (if on line) to be recorded into the file previously designated the RECORD file. If the playback-on function (F1) is active and the terminal is either off line or in the on-line half-duplex mode, data from the playback file may also be recorded under control of F2.

Data recording will be terminated when the operator inputs an F4 (record-off) function.

When operating under control of the F2 function, the terminal will, if the RECORD file is in *line* format, write a record each time the SKIP key is pressed, or if on line, when a receive-end-of-line (RCVEOL) sequence is received. If the RECORD file is in *continuous* format, the terminal will write a record each time the line length specified during the original creation of the file is reached. When printed, the EOL symbol is indicated by the symbol ☒

When the F2 (record-on) function is active, the RECORD indicator lamp will remain lit.

Playback Off — F3. The *playback-off* function causes termination of data playback. One additional character is processed from the playback file after the F3 function is input.

Activating the F3 function will cause the PLAYBACK indicator lamp to extinguish.

Record Off — F4. The *record-off* function causes termination of data recording.

Activating the F4 function will cause the RECORD indicator lamp to extinguish.

Rewind Playback — F5. The *rewind-playback* function causes the internal pointer associated with the currently defined PLAYBACK file to reset to the beginning of the file, in effect “rewinding” the file. Note that two pointers normally are maintained, one for the playback file and one for the record file. When both the PLAYBACK and RECORD functions are designated to the

same file, the two pointers are maintained to be exactly equal at all times. Thus, a *rewind-playback* (F5) has the same effect as a *rewind-record* (F6).

The terminal will indicate execution of F5 by performing a carriage-return/line-feed if the terminal is either off line, or on line in the half-duplex mode. In both cases the printer must be enabled.

Rewind Record — F6. The *rewind-record* function causes the internal pointer associated with the currently defined RECORD file to reset to the beginning of the file, in effect “rewinding” the file. See the discussion of F5 above for special case considerations. The terminal will indicate execution of F6 [if in the half-duplex mode with the printer enabled (F9)] by performing a carriage-return/line-feed.

Playback Forward — F7. The *playback-forward* function causes the current PLAYBACK record to be printed. If the playback file pointer is positioned at the beginning of a record before F7 is input, a complete record will be printed. If the file pointer is positioned past the beginning of the record, the remainder of the record will be printed. These actions are identical for both *line-* and *continuous-*format files. If a *line-*format file, the remainder of text through the end-of-line character will be printed. If a *continuous-*format file, the remaining complete physical record is output. After printing the record, the playback pointer will be moved to the beginning of the *next* record. If the current record is the end of file, nothing occurs.

When the terminal processes a *continuous-*format file, a single F7 entry may cause several lines of printing if a single physical record contains several end-of-line characters.

Playback Reverse — F8. The *playback-reverse* function causes the playback file pointer to reposition to the beginning of the current record. If the pointer is already positioned at the beginning when the F8 function is input, the pointer will move toward the beginning of the file to the beginning of the next previous record.

Printer On — F9. The *printer-on* function enables the terminal printer. The printer is normally ON and is switched ON when power is applied to the terminal and when the CMD key is pressed. The printer will remain ON unless the printer is disabled from the interactive ASR mode (i.e., from the communications line, a RUN file, or entry of an F0 function). The primary purpose of the F9 function is to re-enable the printer after it has been disabled.

Printer Off — F0. The *printer-off* function disables the printer. Any output normally executed by the printer (except for local COMMAND mode inputs: see F9) is ignored. In all other respects, data is processed normally. The F0 function is useful in the ASR mode to copy a file or to transmit data to a host system and a printout of the data is not needed. When the terminal is in the F0 mode (printer-off), the operator may choose to enter F9 mode (printer-on) momentarily to monitor progress of the data transfer. Execution of any command, regardless of the source, will cancel F0 and enable the printer until another F0 is performed.

Special Keys

The $\overleftarrow{\text{CHAR}}$ and $\overrightarrow{\text{FIELD}}$ keys, useful in the EDIT mode, are also operable when the terminal is in the ASR mode. In the ASR mode, if the RECORD function (F2) is not ON, the special keys have no effect. The following discussion concerns use of the terminal off line with the RECORD function ON.

CHAR Key. Pressing the $\overrightarrow{\text{CHAR}}$ key moves the printhead (and the record pointer) one character forward or backward. When the $\overrightarrow{\text{CHAR}}$ key is pressed, the printhead will move to the left, or backspace, one character. If the key is held depressed, the printhead will continue moving to the left until the key is released. The back-character function is referred to as an “intelligent” backspace since the printhead will simply move backward over blank spaces but will perform a line-feed before it backspaces over the first non-blank character position it encounters. This action positions the printhead directly under printed characters to be corrected, or simply moves the printhead over blank spaces which will be filled with data from the keyboard.

The $\overleftarrow{\text{CHAR}}$ key is useful to correct errors in entered data when the terminal is in the ASR mode. To correct an error on the current line of data entered, press the $\overleftarrow{\text{CHAR}}$ key until the printhead is backspaced to the desired location for error correction. The correct information may then be entered from the keyboard. The new data will “write over” old data in memory.

The $\overrightarrow{\text{CHAR}}$ key, in conjunction with the SHIFT key, is useful for moving the printhead (and the record pointer) to the right after making a correction to entered data. When the $\overrightarrow{\text{CHAR}}$ key is pressed while the SHIFT key is pressed and held, the printhead will move to the right one character position. If the $\overrightarrow{\text{CHAR}}$ key is held depressed, the printhead will continue moving to the right until the $\overrightarrow{\text{CHAR}}$ key is released. Note that for multiple

character forward spacing, the SHIFT key need be pressed only for the first character-forward.

When character-forward is active, characters will be printed as they are “passed over”. If a character-forward is attempted at the end of a record, the audible tone will sound and the printhead will remain at its current position; in this event data already entered will not be affected. Any number of forward- and back-character operations are permissible in any particular record.

For files in the *line* format, the $\overrightarrow{\text{CHAR}}$ key is operable only on the current record. For *continuous*-format files the $\overrightarrow{\text{CHAR}}$ key is operable from any point within the file throughout the entire range of the file in either direction.

FIELD Key. In the ASR mode the $\overrightarrow{\text{FIELD}}$ key may be considered as a *record* forward/back key. The direction of the FIELD key is determined by the use of the SHIFT key. The $\overrightarrow{\text{FIELD}}$ key by itself moves the printhead and record pointer beginning of the current record. When the FIELD key is pressed in conjunction with the SHIFT key, the printhead and the record pointer will move to the right (or forward) to the last character of the current record. Like the SHIFTEd $\overrightarrow{\text{CHAR}}$ key, a SHIFTEd $\overrightarrow{\text{FIELD}}$ key will cause all characters “passed over” to be printed.

When corrections are desired toward the beginning of a record, the $\overrightarrow{\text{FIELD}}$ key is often more expedient than multiple $\overrightarrow{\text{CHAR}}$ keys. Likewise, once a correction is made and keyboard entry is desired at the end of the data already entered, the SHIFTEd $\overrightarrow{\text{FIELD}}$ key is useful in passing over all data between the current printhead position and the rightmost character previously entered.

Automatic Device Control Characters

The *automatic device control* (ADC) characters are used in a prompt file to control keyboard input and recording of data. The ADC characters include the four standard control characters for *Case-One terminals* described in *ANSI Standard X3.28*. The four characters control *playback-on*, *record-on*, *playback-off*, and *record-off* (DC1, DC2, DC3, and DC4, respectively). Additionally, several two-character sequences of these control characters are assigned special playback control functions in the Models 763/765. As in the case of the off-line ASR functions, the EDC and DC3 terminal options must be enabled for the DC1, DC2, DC3, and DC4 control characters to function. The ADC control characters and the two-character playback control sequences are discussed in the following paragraphs.

DC1 (Playback-On). The DC1 control character is generated by pressing the **Q** key while pressing and holding the CTRL key. When DC1 is received from the communications line, the terminal will perform the playback-on function if both the EDC and DC1.3 options are ON. When the terminal files are printed (using the CTRL option), DC1 appears as ␣ .

Data from the currently designated playback file is output to the terminal printer (and to the communications line if the terminal is on-line). Data output will continue until one of the following conditions occur:

- The end of file is reached.
- An F3 (playback-off) is entered by the operator.
- A DC3 (playback-off) is received from the communications line — if the DC1.3 and EDC options are both ON.
- A DC3 is processed from the playback file — if the DC3 and EDC options are both ON.

The DC1 character has little practical use within the body of a prompt file, since playback must already be active for the prompt file to print out. In prompt files playback is usually resumed, once stopped, by the operator's use of the ENTER key following entry of data in response to a prompt. The ENTER key function is similar to the ASR mode F1 function when the terminal is off line; i.e., the terminal begins processing the playback file at its last point.

In the case where the terminal is on line and in the full-duplex mode, it should be noted that data output from the memory files under control of a DC1 will not be printed.

DC2 (Record-On). The DC2 control character is entered by pressing the **R** key while pressing and holding the CTRL key. When terminal files are printed (using the CTRL option), DC2 appears as ␣ .

DC2 is used often in prompt files to initialize recording of a field of data. When DC2 is encountered in a playback

(or prompt) file, the terminal RECORD function is immediately activated if the terminal is off line. When the terminal is on line, a DC2 from a prompt file is ignored.

When the terminal receives a DC2 from the communications line, the EDC and DC2.4 options must both be ON to cause the record-on function. ADC characters which are executed are not recorded; for example, a DC3 character received from the playback file is recorded **only** if the EDC or DC3 terminal options are OFF.

All data is recorded regardless of its origin. Data may come from the keyboard, the communications line, or from characters following the DC2 in the playback file. The latter case is the usual means for recording constants or including field delimiters in a prompt file.

DC3 (Playback-Off). The DC3 control character is entered by pressing the **S** key while pressing and holding the CTRL key. When terminal files are printed (using the CTRL option), DC3 appears as ␣ .

DC3 is used to terminate processing of playback data. Processing the DC3 permits the character immediately following to be processed prior to suspension of playback. Typically, the character following DC3 is DC2. This convention causes a prompt to be displayed, playback to be switched off, and the record function to be switched on in anticipation of the operator's response to the prompt.

When the terminal receives a DC3 from the communications line, the EDC and DC1.3 options must both be ON to cause the playback-off function.

DC4 (Record-Off). The DC4 control character is entered by pressing the **T** while pressing and holding the CTRL key. When terminal files are printed (using the CTRL option), DC4 appears as ␣ .

The DC4 is used to terminate recording of data. The recording function is switched off immediately upon encountering DC4.

Creating Prompting Files

The four ADC characters described above are sufficient for basic prompt/response uses. Other control characters, discussed in subsequent paragraphs serve to enhance the effectiveness and versatility of prompt files. It should be noted that a record file, in addition to the prompt file, must be used to record all information gathered by a prompt file.

Recording Constants

To record a constant from the prompt file into the record file, the constant character string should be preceded by DC2 (record-on) and followed by DC4 (record-off). For example, to record **TEXAS INSTRUMENTS** into the record file, the following should appear in the prompt/playback file:

```
%TEXAS INSTRUMENTS%
```

Recording Variables

To record a variable field of data entered by the operator from the keyboard, three ADC functions are used. First, **PLAYBACK** should be disabled to avoid recording constant data from the playback file. Second, recording should be enabled. Third, following input of the variable by the operator from the keyboard, recording should be terminated. It should be noted here that the **ENTER** key has desirable characteristics for use in terminating a prompted field of data entered from the keyboard. When the terminal is neither on-line nor in the **EDIT** mode, pressing the **ENTER** key activates playback in a manner similar to the **F1** function. Thus, to record keyboard-entered variable data **DC3 DC2 DC4** should appear in the prompt/playback file. This sequence will switch playback off after processing the next character (**DC2**) and activate the recording of data. The terminal will then remain in the **RECORD** mode (with the **RECORD** indicator lamp illuminated) until some external stimulus occurs to change this state, and the terminal is ready to accept and record data entered by the operator.

After entering the variable data, the operator should press the **ENTER** key to cause the playback file to again begin processing. The next character encountered in the playback/prompt, **DC4**, will terminate the recording of data.

Simple Prompt/Response

Many prompt applications require simply notifying the operator of the data required and subsequent recording of the operator's response to the prompt. For example, to request and acquire the operator's name, the following could be used in the playback file:

```
ENTER YOUR NAME%: %>%%
```

The constant text string **ENTER YOUR NAME** is displayed on the printer. The colon is both displayed to the printer and recorded to the current record file, and the colon serves as a field delimiter. Playback is then suspended and the record function is activated. Following the operator's keyboard entry to the prompt and pressing the **ENTER** key, recording is terminated and the playback/prompt file continues to the next prompt.

Other Prompt/Playback Control Characters

Additional useful prompt file control characters include the combined ADC characters and horizontal tab. Note that these control characters must be executed from a prompt file.

Horizontal Tab — The *horizontal-tab* control character (**HT**) is entered by pressing the **I** key while pressing and holding the **CTRL** key. When terminal files are printed (using the **CTRL** option), **HT** appears as `%`. The horizontal-tab function is usable off-line only.

The **HT** control character must be entered followed by two numeric characters (no spaces). While called *horizontal tab*, the Models 763/765 **HT** function is actually a means to move or "address" the printhead in an absolute manner. The printhead will move to the left or right to the column specified by the two-digit number following **HT**. Print positions are numbered from left to right, beginning with position 01 and ending at the right margin with position 80. Two digits must always be supplied with **HT**. If the desired position is less than 10, a leading zero must be used (e.g., **HT01**, **HT06**). Recalling the example above of a simple prompt/response sequence with added characters to aid operator legibility:

```
%10ENTER YOUR NAME%: %40%%
```

Note the addition of the two **HT** functions which will cause the printhead first to move to column 10 and then print the prompt. Prior to accepting the operator's response, the printhead will move to column 40.

Multiple Character Playback Functions — There are four two-character special functions available for prompt files: **DC3 DC1**, **DC3 DC3**, **DC3 ESC**, and **DC3 DLE**. These functions may appear at any point within a playback file. The only requirement is that they be logically contiguous.

DC3 DC3 — The dual **DC3** function used at the end of a prompt file will switch off playback and "rewind" the file currently designated the **PLAYBACK** file. This function is operable only when the terminal is off line and the **EDC** and **DC3** options are **ON**.

DC3 DC1 — This dual function “rewinds” the current PLAYBACK file, but does not switch off the playback function. Instead, playback continues from the beginning of the playback file. The DC3 DC1 function is used at the end of a prompt file when an unknown number of identical prompt/response sequences are desired. The resulting “looping” may be exited by the operator momentarily pressing the ESC key in conjunction with the FCTN key. This scheme assumes the existence of a *RUN file*. Otherwise, a playback loop may be exited by using the ASR mode F3 function (playback-off). The DC3 DC1 function operates only when the terminal is off line and the EDC and DC3 options are ON.

DC3 DLE — The DLE character (data-link-escape) is entered by pressing the P key while pressing and holding the CTRL key. When terminal files are printed (using the CTRL option), DLE appears as `␣`.

The DC3 DLE function is operable only when the terminal is off line and both the EDC and DC3 terminal options are ON. The DC3 DLE sequence passes on the DC3 function for further processing without affecting the state of the playback file (the DLE character is ignored by the terminal). When the terminal is on line, the characters are processed individually. The DC3 DLE sequence is necessary to record a DC3 character in a data file while performing off-line prompting operations. Recording of DC3 characters in a data file during prompting operations becomes important when establishing an ASR protocol in a standard ASR telecommunications environment.

When operating in a standard ASR telecommunications environment (see Section III), the DC1.3 function may be used to establish a standard ASR protocol. In this mode the host system requests transmission of a record

by sending to the terminal a DC1. The terminal will in turn transmit data from the playback file until a DC3 is processed. By recording DC3 characters to the record file during data acquisition, the prompt file may format the data file for later transmission to the host system. Other means of acquiring data stored in the terminal are available to the host system: For these methods please refer to Section III of this manual.

An example of DC3 DLE use may be found in an order-taking application in which an unknown number of part numbers and quantities are entered by the operator. The following might appear in the playback file:

```
PART#: 4420554400TY: 4450555146456
```

Some items in the above prompt file may be familiar from the previous discussions. Beginning with DC3 DLE the following actions are taken: DC3 DLE causes a DC3 character to be written to the record file to act as an ASR control character for future dialog with a host system (or other data collection device). CR LF causes a new line to be initiated on the printer, and DC3 DC1 causes the file to “rewind” and be processed again.

Assuming the existence of a RUN file, the operator may terminate this playback loop by pressing the ESC key while pressing and holding the FCTN key. If return to a RUN file is not desired, the loop may be exited by using the ASR mode F3 function (playback-off).

DC3 ESC — Usually placed at the end of a prompt file, DC3 ESC causes termination of the playback function and returns to the last point of an execution of a RUN file.

Creating RUN Files

In many instances the central controlling entity of a Models 763/765 application is one or more RUN files. A RUN file must be created as a *line*-formatted file. Contents of a RUN file may consist of:

- Any command (defined in Section II) which is also legitimate for operator entry
- Any of the ten ASR functions
- Character string constants.

When a RUN file is executing, the commands and ASR functions contained within the file will not be output to

the printer. Character string constants may be displayed or otherwise processed, depending on the state of the terminal and the context in which the character strings appear. A RUN file is activated via the RUN command, defined in Section II of this manual.

RUN File Commands

A command appearing in a RUN file must conform precisely to the syntax rules for the command:

- Only one command may appear on a line.
- The command must begin in column one of the line in which it appears.

- Other data may follow the command, separated by at least one space. If any other such data is included in the line, it is treated strictly as commentary.

Since RUN files must be in *line* format, the use of commentary will not affect system performance (except, of course, during transmission of the RUN file via the communications line) and may simplify later modification to the RUN file. Commands appearing within a RUN file are not output to the printer.

ASR Functions in RUN Files

Each of the ten ASR mode functions (as defined for keyboard input in this section) may also be used in a RUN file. An ASR function is specified using two characters, the first of which must always be "#". The second character must be a digit from 0 through 9 corresponding to the ASR function desired. For example, a RUN file line to rewind playback, rewind record, and switch on playback would appear as **#5#6#1**.

Unlike commands, any number of ASR functions may appear on a single line within a RUN file. The "#" of the first ASR function on a line must be in column one. Subsequent functions on the same line must be concatenated with the first function with no blank or other characters interspersed within the string of ASR functions. Any data appearing in a line following ASR functions will be treated as a character string constant (see below).

In addition to the ten ASR mode functions, one other function is available for use in a RUN file: the **#ESC** characters. This function causes control of the terminal to exit the ASR mode and resume RUN file operation. It is generally used to terminate operator response to a RUN file prompt instead of the SKIP or ENTER key. The **#ESC** function also may be used *within* the RUN file to cause execution to resume immediately after an opera-

tor response. This function is further detailed later in this section.

Character String Constants

Characters which appear in RUN files, except as two-character ASR functions, the text of a command, or the commentary following a command, are treated as *constants*. The constants may appear immediately following a # character. In fact, when constants for display are specified in this manner, the processing time of the RUN file is decreased. Comments should not be placed on lines which contain ASR functions.

The terminal treats the string constants in accord with the state of the terminal and the context in which the constants appear. Basically, they are simply constants which can be directed for output to the printer or the current record file if the terminal is off line, and also to the communication line if the terminal is on line. Processing is accomplished by the ASR function processor firmware if the "#" appears at the beginning of the string. Processing time is decreased using the # prefix to the string because the string will not first be examined by the terminal to determine if it is a command. Assuming that the terminal is off line with the printer enabled, these strings (any ASR functions to the contrary) will be output to the printer as the RUN file is processed.

Using ASR functions the constant can be recorded to the current RECORD file without being printed. For example, to record the string **T1765** without printing it, the following line might be used: **#0#2T1765#4**. To output the string to the printer, assuming the printer is enabled, the RUN file line would simply contain **T1765**.

Character string constants may contain normally nonprintable characters: Carriage return, line feed, and horizontal tab are typical nonprintable characters frequently used in RUN files for data collection and other applications.

Using the RUN File

The RUN file concept affords the applications programmer a means for accomplishing automatically a wide range of tasks within the terminal. With a well designed library of RUN files, the end user of the terminal need only know how to activate a RUN command and type responses. Any other functions necessary to accomplishment of the application can be done by one or more RUN files operating in conjunction with prompt files.

Activating a Prompting File

A common use of a RUN file is to set up data files for subsequent use of a prompt file and then to activate the prompt file. The basic steps to implement such a function are to assign playback and record functions to the specific files desired via COMMAND lines within the RUN file and then using an ASR function to activate playback.

For example, suppose a data acquisition prompt file named **COLECT** requires using a temporary file for output that is in *line* format. The temporary file must contain at least 25 lines and each line is 80 characters in length. A RUN file to set up the temporary file and activate the prompt file might appear as follows:

```
CREATE TEMP L 25 80          TEMPORARY STORAGE
CHANGE RECORD TO TEMP      ASSIGN RECORD FILE
CHANGE PLAYBACK TO COLECT  PROMPT FILE
#5#6#1
```

The effect of this RUN file is to

(line 1) create a *line*-formatted file of 25 eighty-character records called **TEMP**

(line 2) assign the RECORD function to the newly created file **TEMP**

(line 3) assign the PLAYBACK function to the prompt file named **COLECT**

(line 4) rewind **COLECT**, rewind **TEMP**, and activate playback.

Once a prompt file is active, control may be returned to the RUN file in any of three ways:

1. If the EDC and DC3 options are both ON, the control sequence **DC3 ESC** encountered in the playback file will deactivate playback and reactivate the RUN file on the line immediately following the line in the RUN file which activated the prompt file.
2. The operator may terminate playback and reenter the RUN file by pressing the ESC key in conjunction with the FCTN key. This is the normal manner to terminate an iterative playback file operating under control of a **DC3 DC1** combination (rewind playback and continue). For this operation to function smoothly, a **#ESC** combination should be inserted in the RUN file at a point where it will be immediately executed upon reentry into the RUN file.
3. The communications line will terminate a prompt file and cause reentry into the RUN file (provided the terminal is on line) with the sequence **ESC<**.

In all three cases above the playback pointer remains in the position last occupied. Thus, if a RUN file contains a second **#1** with no intervening rewind or redefinition of

the playback file, the playback file will resume operation at its last point of execution prior to suspension.

Reentering a RUN File from a Prompting File.

Recalling the above example for activating a prompt file, suppose the prompt file is to operate on a temporary file. After a data collection transaction with the operator, the terminal is to transfer the data into a more permanent data file. This technique is often used to afford a higher degree of integrity to the data collected.

The following RUN file defines a temporary file, activates a prompt file, and, following reentry, concatenates the recently acquired data onto the end of a larger, more permanent file.

Commands	Comments
OFFLINE	TAKES TERMINAL OFFLINE
CHANGE EDC TO ON	ACTIVATES EDC
CHANGE DC1.3 TO ON	ENABLES PLAYBACK ADC
CHANGE DC2.4 TO ON	ENABLES RECORD ADC
CREATE TEMP L 25 80	CREATES TEMPORARY FILE
CHANGE RECORD TO TEMP	RECORD FILE IS TEMP
CHANGE PLAYBACK TO COLECT	PROMPT FILE IS COLECT
#5#6#1	
FREE DATA	UNLOCKS DATA FILE
COPY TEMP TO DATA END	APPENDS TEMP TO END OF DATA
LOCK DATA	PROTECTS DATA
DELETE TEMP	DELETES TEMP STORAGE FILE

Through the line **#5#6#1** the function and meaning of each line of the RUN file should be clear. When **COLECT** terminates with the sequence **DC3 ESC**, the RUN file resumes execution with the line **FREE DATA**. The permanent storage file **DATA** is normally **LOCKed** to protect its contents from inadvertent changes. **FREE DATA** permits modification to the **DATA** file. The next line copies the contents of **TEMP** onto the end of the **DATA** file. Recall that the **COPY** command may be used with files of differing formats. It is quite possible that **DATA** was created upon system initialization as a *continuous*-formatted file to conserve memory space.

The next line, **LOCK DATA**, restores protection to the **DATA** file, and the last line of the RUN file deletes the temporary storage file **TEMP** from the terminal catalog.

Terminal Configuration RUN Files

A RUN file also is useful simply for terminal configuration. Environments in which connection into more than one telecommunications network is necessary can employ a RUN file to greatly simplify the task of terminal reconfiguration. Within the RUN file configuration parameters from the **COMMAND** repertoire can be stored in advance.

For example, suppose interconnection to a system called *TSO* is desired. *TSO* operates at 300 baud, half-duplex, even parity, and anticipates each input line to terminate with **DC3**. In addition, the operator desires to use the Model 763/765 **BUFFER** option to enhance throughput. *TSO* normally transmits a **DC1** as a ready prompt. A **RUN** file to configure the Models 763/765 to operate in this environment could be

```
PLEASE DIAL 821-7777
CHANGE ABM TO "LOGON SMITH/33752" S
CHANGE DUPLEX TO HALF
CHANGE SPEED TO 300
CHANGE INTERFACE TO INTERNAL
CHANGE PARITY TO EVEN
CHANGE PCHECK TO ON
CHANGE BUFFER TO ON
CHANGE XMTEOL TO DC3
CHANGE RCVEOL TO CRLF
CHANGE DC1.3 TO ON
CHANGE EDC TO ON
ONLINE
```

The above **RUN** file would configure the terminal for operation in the *TSO* system environment. The line beginning *PLEASE DIAL* displays to the operator the telephone number to dial for connection to the host system. The *CHANGE ABM* line creates a log-on for the system (note that the *ABM* message is protected). The log-on will be transmitted when the operator inputs the *HERE IS* (CTRL 1).

Note that the **EDC** and **DC1.3** options are **ON**. This permits immediate access to a designated playback file immediately upon connection if the host transmits **DC1**. It may be desirable to *change DC1.3* and **EDC** options to off if a conversational mode of operation is desired.

After creation of the **RUN** file, all that is required of the operator to configure the terminal for operation in the *TSO* environment is to enter the command **RUN TSO**. The terminal will respond with **PLEASE DIAL 821-7777**, and once the terminal is internally configured in accord with the **RUN** file, the terminal will print **DONE**. Because of the *ONLINE* command, the **ONLINE** indicator lamp will come on when the **RUN** file is completed.

Data Collection Within a RUN File

Allowable statements within **RUN** files include the **ASR** functions. Although used somewhat differently than within prompt/playback files, **ASR** functions can be included in **RUN** files to implement prompt/response data collection. The rules for such **RUN** files are

1. **ASR** functions must be the first item of a line.
2. Lines containing **ASR** functions may contain only **ASR** functions and character string constants.

3. **RUN** file commands and other functions which permit operator input from the keyboard should begin on a new line. **RUN** file lines which contain **ASR** functions should be terminated by **#ESC**.
4. Throughput is enhanced if string constants used for prompting begin with **#** which causes the remainder of the line to be passed to the **ASR** processor.
5. The operator must signal termination of input by pressing the **ESC** key in conjunction with the **FCTN** key. Record and playback functions will terminate automatically when the **FCTN/ESC** key sequence is used to return to a **RUN** file; therefore, use of **#4**, etc., to terminate recording of keyboard data is not necessary.
6. The **SKIP** key, used during **RUN** file entry, has no effect during execution of the **RUN** file. Carriage control must be explicitly stated.

As an example, assuming the terminal is configured for proper operation, the following is a simple data collection **RUN** file:

```
#0#2//#4#9#5
#4#5
#4#5
```

Note that commentary is not included in the above example. Since no operator commands appear in the file, any commentary would be output. The first line causes two slashes, used as field delimiters, to be recorded to the current record file. Since **#0** (printer-off) precedes **#2** (record-on), the slashes will not be output to the terminal printer. The **#9** (printer-on) reenables printing after recording is done. Carriage control is imbedded in the first (and subsequent) prompt since the **SKIP** key, which records **EOL**, is not recognized. The **#2** function appears at the beginning of inputs, causing recording a colon as a field delimiter. The **#4**, which terminates the recording, begins the line following the line which initiated recording. Note that the line is terminated by a **#ESC**. The **FCTN/ESC** keys entered by the operator will cause the **#4** to be processed. The **#ESC** will then cause resumption of the **RUN** file.

SECTION VI

TYPICAL APPLICATIONS

The Models 763/765 memory data terminals are capable of a variety of applications, ranging from simple recording and playback of data to remote control of the terminal by a host system. The complexity of an application is primarily determined by the degree of interaction desired between the operator and the data terminal. Sample applications in this section include an operator prompt file, a RUN file, a complete operations system, and an application illustrating remote control of the terminal by a host computer. The sample applications in this section are fully developed and easily implemented, but may require modifications to meet your specific data collection requirements.

Prompting Files

Operator prompt files help organize and standardize data collection by ensuring that data is entered and stored in a predefined sequence and that important data fields are not left out. With the Models 763/765 ter-

minals a file can be created to fulfill the following capabilities:

- Operator prompts
- Automatically switch record on and off
- Automatically switch playback off
- Automatically rewind and restart playback
- Automatically insert field delimiters.

Prompting files are easy to implement: Simply create a file and type in the desired prompts and control characters.

Sample Order Form

For example, an application in which orders must be collected and stored must first define an order form format. The following form is an example of a simple prompting order form.

TEXAS INSTRUMENTS INC.
PO BOX 1444
HOUSTON, TEXAS 77001

ORDERED BY

NAME : -
STREET: -
CITY : -
STATE : -
ZIP : -
PHONE : -

ITEM	QTY	UNIT PRICE
?	*	\$
?	*	\$
?	*	\$
?	*	\$
?	*	\$
?	*	\$

In this order form the operator is required to separate the information stored for ORDERED BY with a "-", precede all item numbers by "?", precede all quantity numbers with "#", and precede all unit price numbers by "\$".

Order Form Implementation

1. To implement the order form, we first CREATE a *continuous-format* record file called **ORDERS**:

```
▶ CREATE ORDERS C 5 80
▶ CG RECORD TO ORDERS
```

2. Press FCTN 2 to switch record on and type the following, ending each line by pressing the SKIP key:

```
730TEXAS INSTRUMENTS INC.
730PO BOX 1444
730HOUSTON, TEXAS 77001
74708ORDERED BY
7NAME : 2-9999
STREET: 2-9999
CITY : 2-9999
STATE : 2-9999
ZIP : 2-9999
PHONE : 2-9999

```

```
708ITEM7270TY748UNIT PRICE
708?9999727?9999748$9999
708?9999727?9999748$9999
708?9999727?9999748$9999
708?9999727?9999748$9999
708?9999727?9999748$9999
708?9999727?9999748$9999
```

3. Press CMD and type EDIT ORDERS. Then use the PRINT function (F4) to print the contents of the file. The contents of the file printed by the editor should read:

```
▶ ED ORDERS
+
#730TEXAS INSTRUMENTS INC. #730PO BOX 1444#730HOUSTON, TEXAS 77001#74708ORDERED
D BY#NAME : 2-9999#STREET: 2-9999#CITY : 2-9999#STATE : 2-9999#ZIP : 2-9999
#PHONE : 2-9999##708ITEM7270TY748UNIT PRICE#708?9999727?9999748$9999# 708?
9999727?9999748$9999#708?9999727?9999748$9999#708?9999727?9999748$9999#
708?9999727?9999748$9999#708?9999727?9999748$9999#708?9999727?9999748$9999#
#00000000000000000000000000000000
ETX
+
```

How ORDERS Form is Generated

The HT control character, plus two numeric characters, set horizontal tabbing for header information and for printhead alignment for operator input. The technique is the same for each indicated line. Note that the word **NAME:** will be printed but will not be recorded since RECORD has not been switched on. Next, D2 is entered to switch on record followed by the hyphen which is printed and recorded. The D4 then switched record off to prevent the D3 character from being recorded. The terminal will respond to the D3 character by stopping playback and processing one more character: D2. At this point the terminal is idle, waiting for the operator input with record on.

After the operator types a response, pressing the ENTER key starts playback with the next character, D4, which in turn switches record off and prints the next prompt. This method is repeated for each prompt. With prompting file control, characters can be transferred from the playback file to the printer and/or to the record file. The symbol \boxtimes in the editor printout is the end-of-line (EOL) symbol which is placed in the file when the SKIP key is pressed. The EOL will cause a line feed and carriage return when playback locally.

RUN Files

To simplify the operator's task pre-stored terminal setup commands can be executed at will. This capability can be used for:

- Reducing operator manipulation
- Access to several systems with different communications parameters
- Collecting data using selectable predefined forms
- Storing data in separate data files
- Changing from off-line to on-line operation.

The Models 763/765 provide the RUN command to enable the operator to execute a series of prestored commands. A RUN file called *ORD* would be activated simply by pressing CMD and typing:

▶ RUN ORD

The following tasks can be accomplished using RUN files:

- Automatically create, erase, delete, lock, and free files
- Automatically assign files for playback operation
- Automatically assign files for record operation
- Automatically execute ASR functions (record on/off, playback on/off, and rewind record/playback)
- Automatically set configuration parameters
- Automatically switch on and off line
- Print operator instructions.

RUN File Implementation

The following RUN file, named *ORD*, will collect data into a file with prompts for the operator.

```
OF
CG PLAY TO ORDERS
ERASE TEMP
#PRESS "ENTER" AFTER EACH ENTRY
#PRESS "FCTN ESC" TO TERMINATE PRESS "FCTN ESC"
#5#6#1
FREE DATA
CP TEMP TO DATA END
CG PLAY TO
#*****SELECT NEW RUN CMD*****
ON
```

The file will be executed when the operator presses CMD and types the following:

▶ RUN ORD

How the ORD File Works

The above RUN file will automatically set the terminal to off-line (required to operate a prompting file), assign playback to a file named ORDERS (the prompting file discussed earlier), and erase a file called TEMP which is created simply as a temporary data collecting file.

Next, the terminal will printout operator instructions, rewind both the record and the playback file and start playback of the prompting file. At this point, the prompting file will control operation until the operator presses the FCTN and ESC keys to return control to the RUN file. The RUN file then FREES a file called DATA (used to accumulate all the data collected) adds the data from the file named TEMP to the data in the file called DATA. The playback function then is assigned to "nothing" to prevent access to the ORDERS file, other instructions to the operator are printed, and the terminal returns to on-line operation.

With this control, the operator has to remember only one command — "RUN ORD".

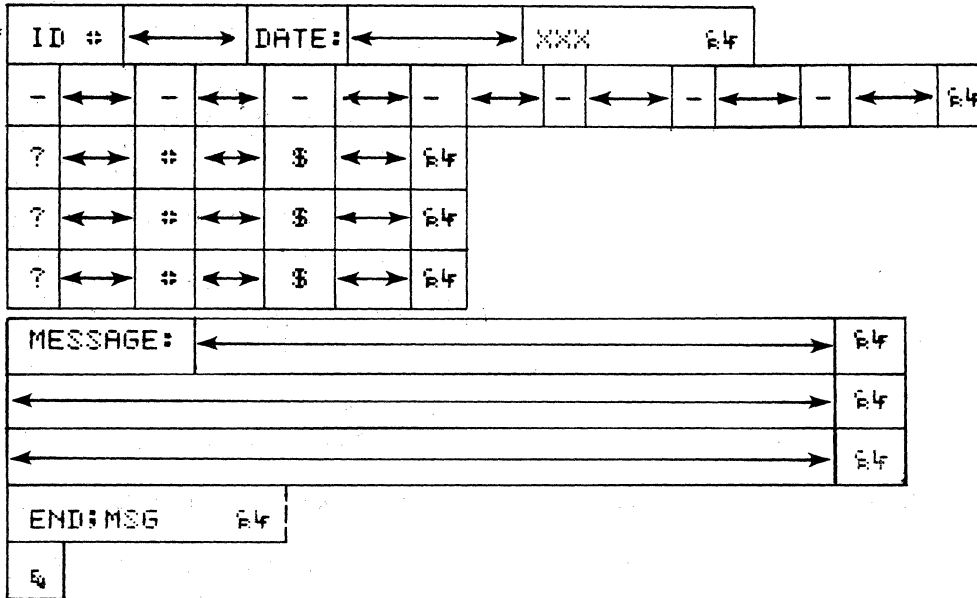
The following special conditions apply to RUN files:

- RUN files must be in *line* format.
- Only one command per line is permitted.
- A RUN command cannot be executed from a RUN file.
- Control can be returned to a RUN file from a playback file by using a DC3 ESC sequence in the playback file.
- The # symbol is interpreted as the function key in a RUN file. If a # symbol is encountered, the terminal converts to the ASR mode.
- To return from the ASR mode to the next run line, the # ESC is recorded in the RUN file.

Complete Operating Systems (To Minimize Operator Interaction)

Some applications require a complete operating system to define an operator's task for an entire day. A host system frequently has specific requirements which determine how data is formatted for processing. Data field

delimiters, header records, trailer records, and special control characters may be required in addition to the data the operator must supply. For purposes of a working example, the following data format for orders and messages will be used as host computer requirements:



The ←→ fields indicate operator input fields. The data can have order information and message information interleaved. To satisfy the requirements shown above (field delimiters, header and

trailer records, and special control characters), a combination of RUN and prompting files are needed. The catalog of a sample operating system may appear as follows:

▶ CATALOG

MEMORY CATALOG

NAME	TYPE	MAXIMUM RECORDS	CHARS/RECORD	COMPLETE RECORDS
START	LINE.L	22	80	19
HEADER	CONT.L	1	36	1
ORD	LINE.L	14	80	14
MSG	LINE.L	17	80	15
LIST	LINE.L	9	80	5
MESSAGE	CONT.L	2	48	2
XMIT	LINE.L	8	80	7
COMMAND	CONT.L	3	36	3
HELP	LINE.L	4	80	2
ORDERS	CONT.L	3	80	3
DATA	LINE	72	80	0
TEMP	LINE	60	80	0
ITEMS	CONT.L	1	80	1

MEMORY AVAIL = 37 80-CHAR LINES
 RECORD FILE: TEMP
 PLAYBACK FILE:

DONE

This system consists of:

Five Prompting Files	Six Run Files	Two Data Collection Files
HEADER	START	DATA
MESSAGE	ORD	TEMP
COMMAND	MSG	
ORDERS	LIST	
ITEMS	XMIT	
	HELP	

The operator would use the Model 763 or 765 during the day to collect message and order information and then transmit the collected data at the end of the day to a host computer for processing.

How the Sample Operating System Works

The operator starts the day by initiating a RUN START command. The START file contains the following:

```

OF
CG EDC TO ON
CG DC3 TO ON
FREE DATA
DELETE DATA
CF DATA L 72
CG PLAY TO HEADER
CG RECORD TO TEMP
DELETE TEMP
CF TEMP L 60 80
%PRESS "ENTER" AFTER ID# & DATE
#5#1
#0#1
#9#E
CF TEMP TO DATA
LOCK DATA
%USE "CMD RUN" AND
%SELECT INPUT: ORD OR MSG

```

The START file sets up certain terminal parameters required for operation. The EDC and DC3 function must be switched on to operate a prompting file. The DATA file is freed and deleted to clear all data previously stored. The DATA file was created to provide a data collection file, regardless of the previous status of the terminal. The prompting file HEADER is set to playback. The next three statements set up a file named TEMP to collect data to be recorded. This file is used for temporarily storing data. With the use of a temporary file, it is possible for the operator to press the CMD key and

erase the complete input without affecting previously collected data. Operating instructions are printed, the HEADER file is rewound, and playback is started.

HEADER File. The operation is then controlled by the HEADER prompting file which is listed below.

```

%ID: %%  DATE: %%%%%E%XX
%%

```

The header file prints the prompts and records the operator response for **ID#** and **DATE** and automatically returns to the START run file, switches off the printer, returns to the prompt file HEADER, records three X's, returns back to the run file, switches the printer back on, copies the TEMP file to the permanent DATA file, and prompts the operator to select another run file.

At this point, the operator has the choice to enter order or message data; to enter orders, the operator presses CMD and types:

```

▶ RUN ORD

```

ORD File. The ORD file is shown below:

```

OF
CG PLAY TO ORDERS
ERASE TEMP
%PRESS "ENTER" AFTER EACH ENTRY
%*****TO TERMINATE PRESS "FCTN ESC"****
#5#6#1
CG PLAY TO ITEMS
#5#1
FREE DATA
CF TEMP TO DATA END
LOCK DATA
CG PLAY TO
%*****SELECT NEW RUN CMD***
ON

```

The ORD file sets the terminal to collect data under control of the prompting files ORDERS and ITEMS, giving operator instructions, collecting data in the TEMP file and copying the collected data into the permanent DATA file. Instructions for the next operation are also given to the operator.

ORDERS File. The ORDERS file described earlier is here divided into two files, ORDERS and ITEMS, to further simplify operator data entry. Previously, if the operator needed to enter more than six items, all header lines would be reprinted. Now, the ORDERS file contains only the header lines as shown below.

```

%30TEXAS INSTRUMENTS INC.
%30PO BOX 1444
%30HOUSTON, TEXAS 77001
%%%08ORDERED BY
%NAME : %-%%
%STREET: %-%%
%CITY : %-%%
%STATE : %-%%
%ZIP : %-%%
%PHONE : %-%%
%
%08ITEM%270TY%48UNIT PRICE
%

```

The ORDERS file is terminated with a DC3 ESC, returning control back to the run file ORD. The run file then changes playback to ITEMS which is a one-line file shown below.

```

%08%7%4%27%#%4%48%$%
%

```

The ITEMS file is terminated with a DC3 DC1 which automatically restarts the playback file for entry of the next line of data. The operator terminates the order entry session by pressing the FCTN and ESC keys simultaneously.

MSG File. The MSG file is as follows:

```

OF
CG PLAY TO MESSAGE
ERASE TEMP
#5#6#1
#0#1
#9#%
FREE DATA
CP TEMP TO DATA END
LOCK DATA
CG PLAY TO
%%***SELECT NEW RUN CMD***
ON

```

The MSG file has the same function as the ORD file, except it sets up control for prompting by the MESSAGE file shown below:

```

%%10MESSAGE TRANSACTION
TYPE IN FULL% MESSAGE, %THEN DEPRESS ENTER%% END:MSG
%%

```

This file does local printing for operator instruction, inserting a portion of the message as header data in the TEMP file and, when complete, adds a trailer record.

XMIT File. The XMIT run file sets up the terminal for transmission and transmits the data to an external device. The external device can respond with the remote control ESC < which will return control to the run file and print the message *DATA TRANSMITTED*. This method provides the operator a positive indication the data was transmitted and the external device received the data.

```

LOCK DATA
CG SPEED TO 9600
CG PLAY TO DATA
ON
#0#5#1
%% DATA TRANSMITTED %%

```

All other files in the operating system serve as aids to the operator.

LIST File. The LIST run file provides the operator a copy of the collected data and instructions for making corrections.

```

CP DATA TO PRINTER CTRL
IF INCORRECT EDIT "DATA" FILE

```

```

***SELECT NEW RUN CMD***
ON

```

HELP File. The HELP run file, in combination with the COMMAND prompting files, prints a list of run files which can be accessed by the operator.

AVAILABLE RUN COMMANDS

```

RUN START
RUN ORD
RUN MSG
RUN LIST
RUN XMIT

```

Summary

A carefully designed operating system helps even an inexperienced user collect and transmit orders and messages by knowing only four easy-to-remember commands:

RUN START RUN ORD RUN MSG RUN XMIT

Remote Control by a Host System

The Models 763/765 memory terminals have the capability to be controlled remotely. That is, a run file can be transmitted from a host system to the terminal, automatically loading a complete operating system without operator intervention.

Since the host system can load operating systems, central control of inputs from remote field operation is possible. Such a system can provide the means to update user input and host system software processing, so that both occur at the same time. The host can

automatically collect data from the remote terminal and transmit new files, messages, and instructions in one phone call. Another benefit is that data can be loaded into memory at 1200 baud and by remotely executing the COPY (filename) TO PRINTER command, the host can terminate the phone call while the data is printing out locally. This reduces the expense of phone calls and reduces the time load on the host system.

Downloading to the Terminal

The following listing is a host system file which, when transmitted to the terminal, will load the system described in the preceding sections.

```

Control          $00CF START L 22 80
                  $00G RECORD TO START
                  $66526769
                  OF
                  OG EDC TO ON
                  OG DC3 TO ON
                  FREE DATA
                  DELETE DATA
                  OF DATA L 72
                  OG PLAY TO HEADER
                  OG RECORD TO TEMP
Start RUN File   DELETE TEMP
                  OF TEMP L 60 80
                  $P$PRESS "ENTER" AFTER ID# & DATE
                  #5#1
                  #0#1
                  #9#66
                  OF TEMP TO DATA
                  LOCK DATA
                  $P$USE "CMD RUN" AND
                  $P$SELECT INPUT: ORD OR MSG
                  $66467
                  $0LOCK START

Control          $00CF HEADER C 1 36
                  $00G RECORD TO HEADER
                  $66526769
Header Prompt File $ID# 999  DATE:999999999999XXXX
                  9999
                  $66467
Control          $0LOCK HEADER
                  $0CF ORD L 14 80
                  $00G RECORD TO ORD
                  $66526769

```

	OF CG PLAY TO ORDERS ERASE TEMP *PRESS "ENTER" AFTER EACH ENTRY *TO TERMINATE PRESS "FCTN ESC" *5*6*1 CG PLAY TO ITEMS
ORD RUN File	*5*1 FREE DATA CP TEMP TO DATA END LOCK DATA CG PLAY TO *◆◆◆SELECT NEW RUN CMD◆◆◆ ON *6*7 *LOCK ORD
Control	*OCF MSG L 17 80 *OCG RECORD TO MSG *6*5*7*9
	OF CG PLAY TO MESSAGE ERASE TEMP *5*6*1 *0*1 *9*6*
MSG RUN File	FREE DATA CP TEMP TO DATA END LOCK DATA CG PLAY TO *◆◆◆SELECT NEW RUN CMD◆◆◆ ON *6*7 *LOCK MSG
Control	*OCF LIST L 9 80 *OCG RECORD TO LIST *6*5*7*9
	CP DATA TO PRINTER CTRL IF INCORRECT EDIT "DATA" FILE *◆◆◆SELECT NEW RUN CMD◆◆◆ ON *6*7 *LOCK LIST
LIST RUN File	*OCF MESSAGE C 2 48 *OCG RECORD TO MESSAGE *6*5*7*9
Control	*10MESSAGE TRANSACTION TYPE IN FULL MESSAGE, *THEN DEPRESS ENTER* *END:MSG *6*7 *6*7 *LOCK MESSAGE
Message Prompt File	*OCF XMIT L 8 80 *OCG RECORD TO XMIT *6*5*7*9
Control	

	LOCK DATA
	CG SPEED TO 9600
	CG PLAY TO DATA
XMIT RUN File	DN
	#0#5#1
	## DATA TRANSMITTED ##
	##7#0LOCK XMIT
Control	##CF COMAND C 3 36
	##CG RECORD TO COMAND
	##5#7#9
Command Prompting File	##AVAILABLE RUN COMANDS
	##RUN START##RUN ORD##RUN MSG##RUN LIST##RUN XMIT##
	##7
	##0LOCK COMAND
Control	##CF HELP L 4 80
	##CG RECORD TO HELP
	##5#7#9
HELP RUN File	##CF COMAND TO PRINTER
	##7
	##0LOCK HELP
Control	##CF ORDERS C 3
	##CG RECORD TO ORDERS
	##5#7#9
	##30TEXAS INSTRUMENTS INC.
	##30PO BOX 1444
	##30HOUSTON, TEXAS 77001
	##708ORDERED BY
ORDERS Prompting File	##NAME : ##-##-##
	##STREET: ##-##-##
	##CITY : ##-##-##
	##STATE : ##-##-##
	##ZIP : ##-##-##
	##PHONE : ##-##-
	##
	##08ITEM##270TY##48UNIT PRICE
	##
	##7
Control	##0LOCK ORDERS
	##CF ITEMS C 1
	##CG RECORD TO ITEMS
	##5#7#9
ITEMS Prompting File	##08##7##-##-##27##-##-##48##-##-##
	##
	##7
Control	##0LOCK ITEMS
	##CF DATA C 1
	##CF TEMP C 1

Section VI

The concept used transmits remote control commands (explained in Section III) before and after the file data to set the terminal for loading.

Examining the loading for the START file may help explain the technique.

```
ESC OCF START L 22 80
ESC OCG RECORD TO START
ESC 6ESC 5ESC 7ESC 9
```

This sequence creates the START file and assigns it to the RECORD function. The ESC 6 switches on the DC1 through DC4 control of the terminal to enable remote control of record on and off. ESC 5 rewinds the record file, and DC2 switches on record.

ESC 7 switches off the DC1 through DC4 control so that these characters can be recorded and not acted upon by the terminal. The ESC 9 switches off the printer.

After the file data is sent, the following controls are used:

```
ESC 6ESC 7
ESC OLOCK START
```

This sequence switches on the DC1 through DC4 control, switches off record with the DC4 character and again switches off DC1 through DC4.

The next command then locks the file in the terminal to ensure its integrity.

The above procedure is used with each file to be loaded. It should be noted that if the ESC character is to be recorded, the character must be presented twice in the data sent to the terminal as seen in the START file:

```
OF
CG EDC TO ON
CG DC3 TO ON
FREE DATA
DELETE DATA
CF DATA L 72
CG PLAY TO HEADER
CG RECORD TO TEMP
DELETE TEMP
CF TEMP L 60 80
PRESS "ENTER" AFTER ID# & DATE
#5#1
#0#1
#9##
CP TEMP TO DATA
USE "CMD RUN" AND
SELECT INPUT: ORD OR MSG
```

Upon receipt of this sequence, the terminal will record the second ESC character.

SECTION VII

KEYBOARDS AND CHARACTER SETS

National Keyboards and Character Sets

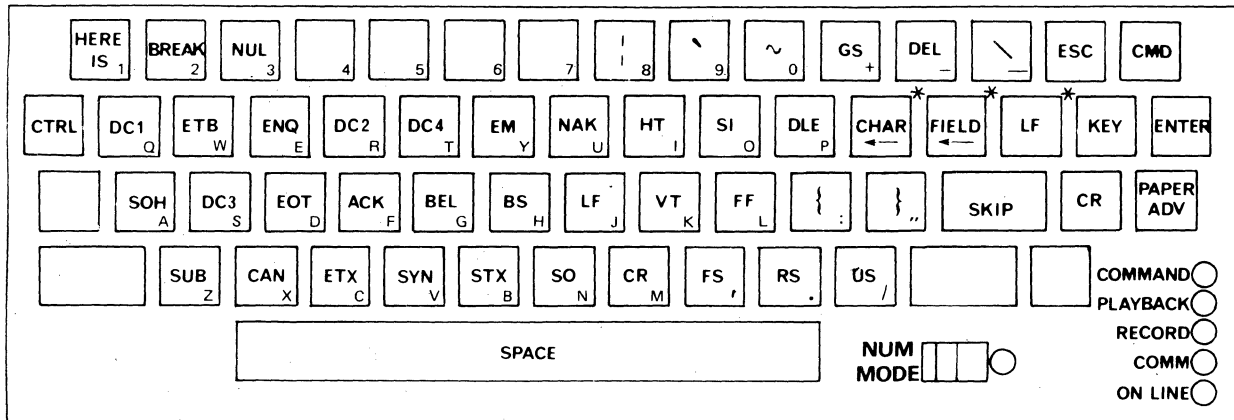
The Models 763/765 terminals can be ordered with any one of six different character sets for the following national alphabets: U.S.A., United Kingdom, France, Denmark/Norway, Sweden/Finland, and Germany. Figure 7-1 illustrates the control character key assignments. Figure 7-2 illustrates the FCTN (function) key assignments, and Figure 7-3 shows the keys active when the NUM MODE switch is selected. Keyboard layout and symbolization for all national keyboards and the key assignments for the available combinations with various mode and shift keys activated are shown in the *Operating Instructions*.

Unless otherwise specified the terminals are shipped from the factory with the keyboard character set corresponding to the country from which the terminals are ordered.

Printer Character Encoding and Generation

The terminal printer generates each character using a five-by-seven dot matrix, electronically heated to transfer to thermally sensitive printing paper. The dot matrix pattern for each printable character is illustrated in an enlarged size in Figure 7-4. Character set encoding, including the bit assignments for all printable characters, control character, and machine action codes (bell, space, line feed, etc.) are listed in Table 7-1.

The following characters are generated/printed by the terminal when the CTRL key is pressed and held.



*The LF key is omitted from the Germany, Sweden/Finland, and Denmark/Norway keyboards; and the CHAR and FIELD keys are each moved to the right one key.

CONTROL CHARACTERS
(From USA Standards Institute Publication X3.4—1968)

ACK	acknowledge	EM	end of medium	NAK	negative acknowledge
BEL	bell	ENQ	enquiry	NUL	null
BS	backspace	EOT	end of transmission	RS	record separator
CAN	cancel	ESC	escape (used for EDC)	SI	shift in
CR	carriage return	ETB	end of transmission block	SO	shift out
DC1	playback ON	ETX	end of text	SOH	start of heading
DC2	record ON	FF	form feed	STX	start of text
DC3	playback OFF	FS	file separator	SUB	substitute
DC4	record OFF	GS	group separator	SYN	synchronous idle
*DEL	delete	HT	horizontal tabulation	US	unit separator
DLE	data link escape	LF	line feed	VT	vertical tabulation

*not strictly a control character

The following ASCII control characters are printed by the terminal:

```

NUL SOH STX ETX EOT ENQ ACK BEL BS HT LF VT FF CR SO SI
DLE DC1 DC2 DC3 DC4 NAK SYN ETB CAN EM SUB ESC FS GS RS US DEL
    
```

Figure 7-1. Control Characters

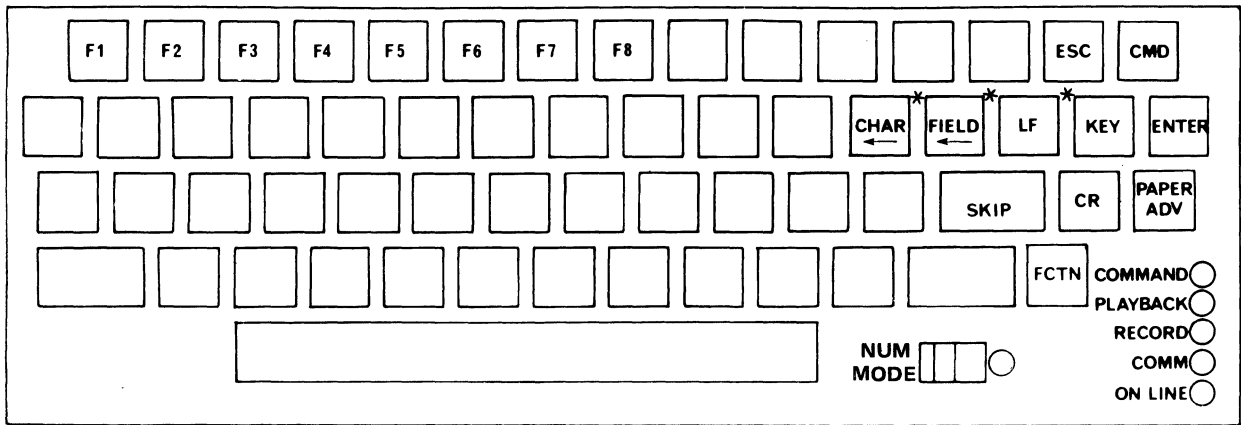
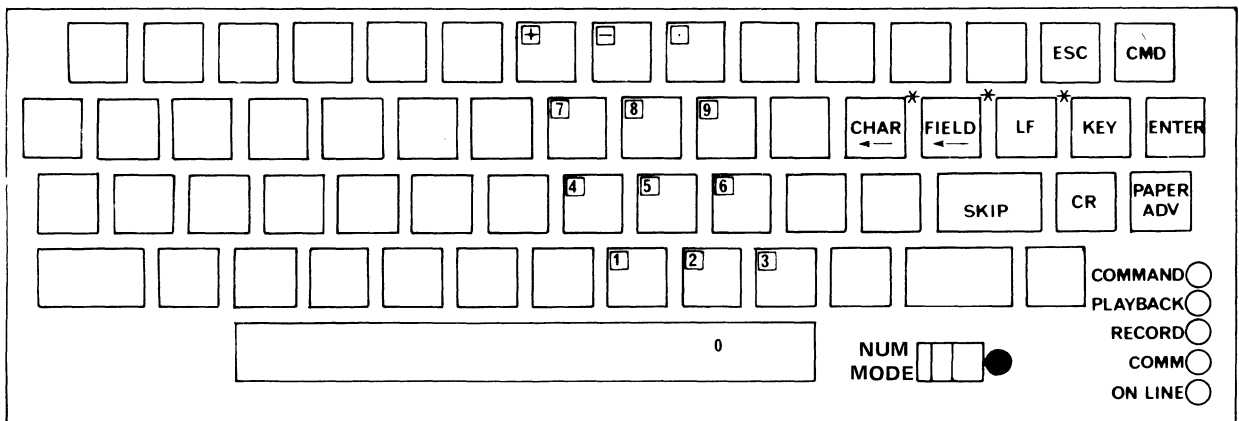
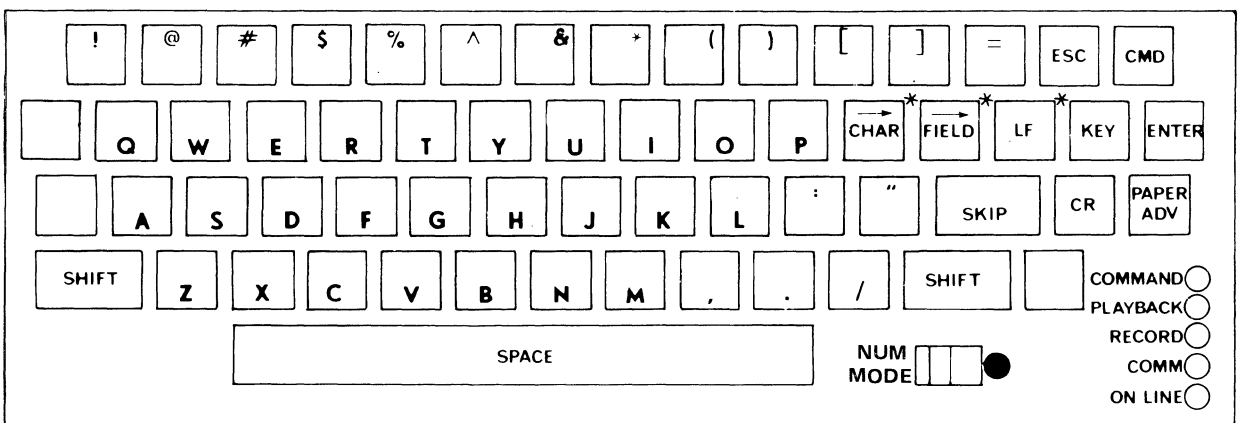


Figure 7-2. FCTN (Function) Key Pressed



NUM (Numeric) MODE Switch Selected, SHIFT Key Not Pressed



NUM MODE selected and SHIFT Key Pressed.

NOTE

Various national characters differ from the above drawing, but the corresponding keys are active.

Figure 7-3. NUM MODE Key Assignments

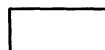
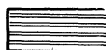

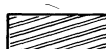

*The LF key is omitted from the Germany, Sweden/Finland, and Denmark/Norway keyboards; and the CHAR and FIELD keys are each moved to the right one key

		CONTROL CHARACTERS				UPPERCASE		LOWERCASE		NATIONAL CHARACTERS	
BITS	b ₇	0	0	0	0	1	1	1	1		
	b ₆	0	0	1	1	0	0	1	1		
	b ₅	0	1	0	1	0	1	0	1		
	b ₄	0	1	2	3	4	5	6	7		
	b ₃										
	b ₂										
	b ₁										
0	0	0	0	0	0	0	0	0	0		
		NUL	DLE	Space							
0	0	0	1								
		SOH	DC1								
0	0	1	0								
		STX	DC2								
0	0	1	1								
		ETX	DC3							1	United Kingdom Character
0	1	0	0							2	German Characters
		EOT	DC4							5	
0	1	0	1							2	Sweden/Finland Characters
		ENO	NAK							5	
0	1	1	0							2	Norway/Denmark Characters
		ACK	SYN							5	
0	1	1	1							3	German Characters
		BEL	ETB							6	
1	0	0	0							3	Sweden/Finland Characters
		BS	CAN							6	
1	0	0	1							3	Norway/Denmark Characters
		HT	EM							6	
1	0	1	0							4	German Characters
		LF	SUB							7	
1	0	1	1							4	Sweden/Finland Characters
		VT	ESC							5	
1	1	0	0							4	Norway/Denmark Characters
		FF	FS							6	
1	1	0	1							7	German Character
		CR	GS							7	
1	1	1	0							8	Sweden/Finland Character
		SO	RS							8	
1	1	1	1							*	*CONTROL CHARACTER
		SI	US							DEL	

Figure 7-4. Printer Character Set

TABLE 7-1. CODE SYSTEM AND CHARACTER SET FOR U.S.A. AND FRANCE UNITS
(See Notes for Other Nations)

8 4	7 3	6 2	5 1	0 0	0 0	0 0	0 0	0 0	0 1	0 1	0 1	0 1	0 1	0 1	0 1
0 0 0 0	NUL	DLE	SP	0	@	P	'	p							
0 0 0 1	SOH	DC1	!	1	A	Q	a	q							
0 0 1 0	STX	DC2	"	2	B	R	b	r							
0 0 1 1	ETX	DC3	# ¹	3	C	S	c	s							
0 1 0 0	EOT	DC4	\$	4	D	T	d	t							
0 1 0 1	ENQ	NAK	%	5	E	U	e	u							
0 1 1 0	ACK	SYN	&	6	F	V	f	v							
0 1 1 1	BEL	ETB	'	7	G	W	g	w							
1 0 0 0	BS	CAN	(8	H	X	h	x							
1 0 0 1	HT	EM)	9	I	Y	i	y							
1 0 1 0	LF	SUB	*	:	J	Z	j	z							
1 0 1 1	VT	ESC	+	;	K	[²	k	{ ⁵							
1 1 0 0	FF	FS	,	<	L	\ ³	l	⁶							
1 1 0 1	CR	GS	-	=	M] ⁴	m	} ⁷							
1 1 1 0	SO	RS	.	>	N	^	n	~ ⁸							
1 1 1 1	SI	US	/	?	O	_	o	DEL							

	Printable Characters		ASR Control Codes
	Printer Control Characters		Extended Line Control
	Codes Generated and Transmitted by the terminal		

NOTES

- ¹ £ on U.K. units
- ² Å on Germany and Sweden/Finland units; Æ on Denmark/Norway units
- ³ Ö on Germany and Sweden/Finland units; Ø on Denmark/Norway units
- ⁴ Ü on Germany and Sweden/Finland units; Å on Denmark/Norway units
- ⁵ ä on Germany and Sweden/Finland units; æ on Denmark/Norway units
- ⁶ ö on Germany and Sweden/Finland units; ø on Denmark/Norway units
- ⁷ ü on Germany and Sweden/Finland units; å on Denmark/Norway units
- ⁸ ß on Germany units

SECTION VIII

SPECIFICATIONS AND ASSEMBLIES

Specifications

The following specifications apply to both Models 763 and 765 and to both the discrete and nondiscrete bubble memory versions of the terminal.

PRINTER

Method: Nonimpact, electrically heated, 5 × 7 dot matrix thermal printhead, prints on thermographic paper.

Character Set: 95 printable characters in normal mode with an additional 33 ASCII or C.C.I.T.T. control characters printed in edit mode.

Character Size: 2.6 mm × 2 mm (0.105 in. × 0.080 in.)

Line Length: 203.2 mm (8 in.); 2.54 mm character spacing (10 characters per inch); 80 characters per line.

Line Spacing: 4.23 mm (6 lines per inch).

Printing Rate: Up to 30 characters per second.

Paper: TI thermographic printing paper, Part No. 972603, 216 mm (8-½ in.) × 30.5 m (100 ft.); last 3 m (10 ft.) color coded.

Platen: Friction-feed.

Carriage Return and Line Feed: Automatic at column 81; no code is transmitted. The 81st character received is buffered and printed on the next line.

KEYBOARD

Code: ASCII, C.C.I.T.T.; 128 codes generated.

ENVIRONMENT

Temperature: Operating: 10°C to 40°C (50°F to 104°F); Storage: -30°C to 70°C (-22°F to 158°F) without paper; -30°C to 40°C (-22°F to 104°F) including paper.

Magnetic Field: Maximum permissible magnetic field at the bubble memory shield in any direction is 40 oersteds (Oe).

Humidity: Operating: 10% to 90% (no condensation); Storage: 10% to 95% (no condensation).

Shock: Operating: 0 G; Storage: 40 G for 18 msec half sine wave.

Vibration: Operating: 0.5 G, 10 to 50 Hz; Storage: 1.5 G, 30 minutes, 20 Hz.

POWER REQUIREMENTS

Voltage: 90-132 Vrms or 184-264 Vrms.

Frequency: 47-63 Hz.

Power: 150 watts maximum.

PHYSICAL

Size: Width: 391 mm (15.4 in.); Depth: 406 mm (16.0 in.); Height: 139 mm (5.5 in.); Weight: 7.7 kg (17 pounds) including paper.

DATA TRANSMISSION

Method: Asynchronous, serial-by-bit, serial-by-character.

Code: ASCII, C.C.I.T.T.; seven-level, 11 bits per character including parity, start, and two stop bits at 10 characters per second (110 baud); 10 bits per character, including one stop bit at speeds above 10 characters per second.

Mode: Operator-selectable full duplex, half duplex, or half duplex with reverse channel.

Parity: Operator-selectable odd, even, or mark parity; Operator-selectable parity checking.

Received Data Buffering: Character buffering on received data, permitting true 30 characters per second operation (no filler characters required after CR or LF).

Transmitted Data Buffering: Operator-selectable line buffering on data to be transmitted, permitting corrections prior to transmission.

Interface: Operator-selectable; Integral originate-only acoustic coupler or EIA RS-232-C interface with Model 765; dc current loop or EIA interface with Model 763.

Baud Rates: Operator-selectable 110, 200, or 300 baud on internal interface, or up to 9600 baud on EIA interface; throughput limited to 2400 baud.

INTEGRAL ACOUSTIC COUPLER (Model 765 only)

Compatibility: Bell System 103/113 data sets (or equivalent) or C.C.I.T.T. V. 21.

Mode: Originate only.

Modulation: Frequency shift keying (FSK).

Transmit Carrier Frequencies: ASCII Mark: 1270 Hz; Space: 1070 Hz; C.C.I.T.T. Mark: 980 Hz; Space 1180 Hz.

Receive Carrier Frequencies: ASCII Mark: 2225 Hz; Space 2025 Hz; C.C.I.T.T. Mark: 1650 Hz; Space 1850 Hz.

Transmit Level: Adjustable only on U.S. models from -20 dBm to 0 dBm.

Receiver Sensitivity: -38 dBm with full duplex and 300 baud operation; -45 dBm with half duplex and 300 baud operation.

EIA INTERFACE (Optional Both Models)

Optional Auxiliary EIA Interface Kit: Interfaces the terminal to an external device such as a modem. Interface cable is a minimum 1.8 m (6 feet) long terminated with a 25-pin male connector (Cannon DB25P or equivalent).

Signal Levels: Serial interface signal levels are defined by *EIA Standard RS-232-C* as follows:

	-25 to -3 Vdc	-3 to +3 Vdc	+3 to +25 Vdc
Data Signal	Marking	Not Defined	Space
Timing or Control Function	Off	Not Defined	On

dc CURRENT LOOP SERIAL DATA INTERFACE (Model 763 Only)

Maximum Current: 60 milliamps (transmit or receive).

Nominal Current: 20 milliamps.

Maximum Voltage Drop: 3 V (received) and 1.5 V (transmit) while marking.

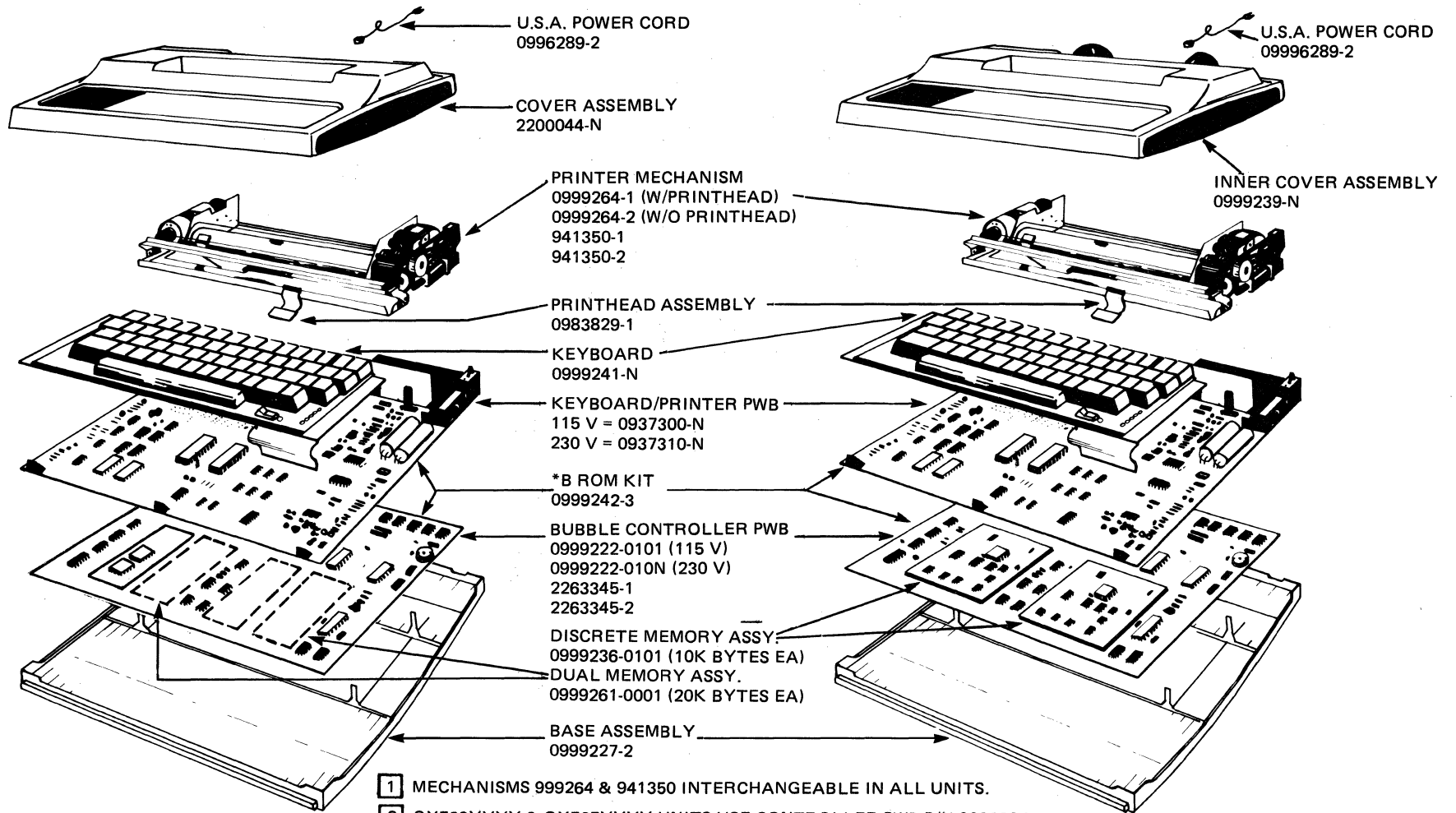
Maximum Transmit Voltage: 50 V while spacing.

Major Assemblies

The major assemblies and their associated part numbers for the Models 763/765 are shown in Figure 8-1.

MODEL 763
2200049-N*

MODEL 765
0999230-N*



64

- 1 MECHANISMS 999264 & 941350 INTERCHANGEABLE IN ALL UNITS.
- 2 OX763YYYY & OX765YYYY UNITS USE CONTROLLER PWB P/N 999222-N;
OX764YYYY & OX766YYYY UNITS USE CONTROLLER PWB P/N 2263345-N.
- 3 DISCRETE MEMORY MODULE ASSY ILLUSTRATED.

*N refers to different dash numbers which denote various models and options. See the *Maintenance Manual* for details.

Figure 8-1. Major Assemblies and TI Part Numbers

USER'S RESPONSE SHEET

Manual Title: 763/765 Systems Manual 2203665-9701

Date of Manual: 4-1-79 Date of this Letter: _____

User: _____ Office/Dept. No.: _____

Company: _____

Street Address: _____

City/State/Zip: _____

Please list any discrepancy found in this manual by page, paragraph, figure, or table number in the following space. If there are any other suggestions that you wish to make, feel free to include them. Thank you.

CUT ALONG THIS LINE

Location in Manual	Comment/Suggestion
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____
_____	_____

FOLD



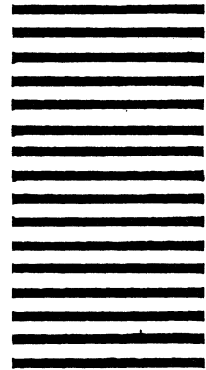
NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 6189 HOUSTON, TX

POSTAGE WILL BE PAID BY ADDRESSEE

Texas Instruments Incorporated
Attn: Technical Publications M/S 7761
P. O. Box 1444
Houston, TX 77001



FOLD

INDEX

ABM (Answer-Back Memory) Option	13	Copying Data:	9
ABM PRT (Print ABM Message)	12	Between Two Files	9
Abnormal Editor Termination	23	From the Keyboard into a File	10
Accessories	4	From a File to the Printer	10
Acoustic Coupler Specifications	25, 63	From Line to Continuous Format	10
Acquiring Data from a Playback File	36	From Continuous to Line Format	10
ADC (Automatic Device Control):		Creating Prompting Files	42
Commands	32	Creating RUN Files	43
Characters	40	Creation of a File	6, 8
ASR Emulation Task	3	Current Loop Interface:	25
Answer-Back Memory (ABM) Option	13	Duplex, Half and Full	28
ASR Function in RUN Files (Also see <i>Functions</i> , <i>ASR Off-Line</i>)	44	Pin Assignments	28
ASR Off-Line Operation	38	(Specifications)	63
AUTOABM (Automatic Answer-Back)	12, 30	Data Collection Within a RUN File	46
AutoABM Trigger	30	Data Set Options, Recommended	28
Automatic Command Execution	23	dc Current Loop (See <i>Current Loop</i>)	25
Automatic Device Control (ADC) Characters	40	DCD (Data Carrier Detect Signal)	29
Buffer:		DC1	32, 41
Option	13	DC1.3 Option	13, 46
Operation	31	DC2	32, 41
Receive, Busy	37	DC2.4 Option	13
Bubble Memory Subsystem	2	DC3 Option	13, 32, 41
Capabilities, Throughput	36	DC3 DC1	43, 45
Carrier Detect Delay	29	DC3 DC3	42
C.C.I.T.T. Interface Definitions	26	DC3 DLE	43
CDT 1000A Data Coupler (Internal Modem):	25	DC3 ESC	43, 45
Pin Assignments	29	DC4	32, 41
CHANGE Commands, Table of	11	Deallocation, File	8
Change Commands Default (Initialized) Parameters	15	Default Parameters	15
CHARacter Key	22, 31, 40	Delay, Carrier Detect	29
Character Sets, Keyboards and	57, 60, 61	DELETE CHARacter Function	18
Character String Constants	44	DELETE LINE Function	17
Characters, ADC	40	Detection of Files	9
Code System and Character Set, ASCII, C.C.I.T.T.	61	Device Service Routines	3
Command Execution:		DSR (Data Set Ready Signal)	29
Automatic (See RUN command)	23	DTR (Data Terminal Ready Signal)	29
From the Communications Line	35	Duplex	
Timing	34	Parameter	12
COMMANDS:		Full-	30
CHANGE	11	Half-	30
COPY	9, 10	Half-, with Reverse Channel	30
CREATE	6, 8	EDC (Extended Device Control) Option	13
DELETE	9	EDC Commands	32
EDIT	15	Editing a Record	22
ERASE	9	Editing, Abnormal Termination of	23
FREE	9	Editing, Text	15
LOCK	9	EIA Interface and Signals	24
RUN	43	EIA (Specifications)	63
STATUS	12	EIA and C.C.I.T.T. Interface Definitions	26
TEST	14	EIA Line Discipline and Timing	29
Commands, EDC	32	Emulation, ASR	3
Commands, File Utility	6	ENQ (Enquiry) Timing	30
Commands, List of	5	Enter Key	10, 19, 20, 21, 22, 42
Commands, Operator Communications Execution		EOTDIS (End of Text Disconnect)	13
Timing	34	Equipment Description	1
Commands, RUN File	43	Erasing Files	9
Commands, Self-Diagnostic	15	ESC (Commands)	32, 33, 56
Communications Line:		FIELD Key	22, 31, 40
Command Execution From	35	File Allocation	6
Recording Data from	35	File, Creation of a	8
Communications Subsystem	2	File Deallocation	8
Complete Operating Systems (RUN Files)	50	File Management System	6
Configuration Parameters	11	Files, Deletion of	9
Constants, Character String	44	Files, Erasing	9
Constants, Recording	42	File Formats (line, continuous)	8
Control Characters	58, 60	Files, Locking/Freeing	9

INDEX

File Utility Commands	6	NUMeric MODE Switch	59
FIND Function (F2)	16	Off-Line Operation, ASR	38
Firmware, Models, 763/765	2	Operating System	3
Freeing/Locking Files	9	Operator Commands: See <i>Commands</i>	
Function Keys (Assignments)	16, 38, 59	Operator Communications Command Execution	
Functions, ASR Off-Line:		Timing	34
Playback-On (F1)	38	Operator Communications Package	3
Record-On (F2)	39	Optional Features	4
Playback-Off (F3)	39		
Record-Off (F4)	39	Page (of a file)	6
Rewind Playback (F5)	39	Parameters, Configuration	11
Rewind Record (F6)	39	Parity Parmeter	12
Playback Forward (F7)	39	PCHECK (Parity Check)	12
Playback Reverse (F8)	39	Peripheral Controller (TMS 8080 Microprocessor)	2, 3
Printer-On (F9)	39	Playback File, Acquiring Data from a	36
Printer-Off (F0)	40	PLAYBACK Option	13
Functions, Edit Mode:		Power Requirements (Specifications)	62
INDEX (F1)	16	PRINT Function	17
FIND (F2)	16	Printer Character Encoding and Generation	57
TOP (F3)	17	Printer Character Set	60
PRINT (F4)	17	Prompt/Playback Control Characters	42
DELETE LINE (F5)	17	Prompting Files:	47
DELETE CHARacter (F6)	18	Creating	42
INSERT (F7)	19	Definition of	38
STOP (F8)	21	Simple	42
Functions, Record Locating	15	Activating a	44
Full Duplex	30	Reentry from, to a RUN File	45
Full Duplex Connections, Model 763	28		
		RCVEOL (Receive End-of-Line) Option	14
Half Duplex:	30	Receive Buffer Busy	37
Connections	28	Record (of a file)	6
With Reverse Channel	30	Record Locating Functions	15
Hardware Interface:	24	Record Lengths at Various Baud Rates, Minimum	37
Acoustic Coupler	25	Recording Data from the Communications Line	35
CDT 1000A Coupler	25, 29	Recording:	
dc Current Loop Interface	25	Constants	42
EIA Interface with an External Data Set	24	Variables	42
EIA Port Interface	24	RECORD Option	13
External Data Set	24	Records (See <i>File Allocation</i>)	5
Internal Modem	25	Reentry from a Prompt File to a RUN File	45
Pin Assignments	Section III	References	iii
Hardware, Models 763/765	1	Remote Control by Host System	53
Horizontal Tab	42	RI (Ring Indicator Signal)	29
Host System Communications with 763/765	32	Run File Commands	43
Host System, Remote Control by	53	Run Files:	
		Creating	43, 48
INDEX Function	16	Complete System	50
INSERT Function	19	Data Collection	46
Inserting Data Within a Record	19	Definition of	38
Inserting Records Within a File	20	Implementation	49
Interface Cables	Section III	Terminal Configuration	45
Interface Definitions, EIA, C.C.I.T.T.	26	Using	44
Interface, Hardware	24		
Interface (Port) Parameter	12	Sample Order Form	47
Internal Modem (CDT 1000A Coupler)	25, 29	SKIP Key	10, 16, 20, 21, 22, 31
		Special Keys	40
KEY Key	13	Specifications	62
Keyboard/Printer	2	Speed Parameter	12
Keyboards and Character Sets	57	Standard Features	3
		STATUS Command	12
Line Receiver Output and Input Voltage Levels	27	Syntax Diagrams:	
Line Disconnect	31	Explained	5
Locking/Freeing Files	9	CREATE Command	8
		COPY Command	9
Main Processor	2	STOP Function	21
Major Assemblies	63	Symbols, Control Character	58, 60
Memory, Bubble	2	System Diagnostics	3
Multiple-Character Playback Function	42		

INDEX

Terminal Configuration Commands	11
Terminal Configuration Run Files	45
Terminal Control, Remote, By Host	53
Terminal Self-Diagnostic Commands:	15
TEST	15
TEST INIT	15
Terminal Utilities	3
Termination Function	21
Timing, Command Execution	34
Text Editing	15
Text Modification Functions	17
Throughput Capabilities	36
Timing, EIA Line Discipline and	29
TMS 8080 Microprocessor Peripheral Controller	2, 3
TMS 9980 Microprocessor Firmware	2
TOP Function	17
Transmit Voltage Level	27
Typical Applications	Section VI
Using the Run File	44
Variables, Recording	42
Voltage Levels, Minimum signal	27
XMTEOL (Transmit End-of-Line) Option	14

Texas Instruments Sales Offices



North American

Arizona: Phoenix	Massachusetts: Boston	Tennessee: Memphis
California: Los Angeles San Diego San Francisco Sunnyvale	Michigan: Detroit	Texas: Dallas Houston San Antonio
Colorado: Denver	Minnesota: Minneapolis	Utah: Salt Lake City
Connecticut: Hamden	Missouri: Kansas City St. Louis	Virginia: Arlington
Florida: Fort Lauderdale Orlando	New Jersey: Newark	Washington: Seattle
Georgia: Atlanta	New York: New York City Rochester	Wisconsin: Milwaukee
Illinois: Chicago	North Carolina: Charlotte	Canada
Iowa: Des Moines	Ohio: Cleveland Dayton	British Columbia: Vancouver
Indiana: Fort Wayne Indianapolis	Oklahoma: Tulsa	Ontario: Ottawa Toronto
	Pennsylvania: Philadelphia Pittsburgh	Quebec: Montreal

International

Argentina: Buenos Aires	Germany: Essen Frankfurt Freising Hamburg Munich
Australia: Melbourne Sydney	Holland: Amstelveen
Austria: Vienna	Italy: Milan Rome
Belgium: Brussels	Japan: Osaka Tokyo
Denmark: Copenhagen	Norway: Oslo
England: Bedford Cheshire Slough	Republic of Singapore: Singapore
Finland: Helsinki	Spain: Barcelona
France: Lyon Nice Paris	Sweden: Stockholm

Texas Instruments reserves the right to change its product and service offerings at any time without notice.



TEXAS INSTRUMENTS
INCORPORATED