

TENET USERS MANUAL
EXECUTIVE

DEC 28 1970

EXECUTIVE COMMAND SET

Subsystem Call Commands

BASI[C]	Calls the TENET BASIC compiler	p.17
FORT[RAN]	Calls the FORTRAN IV compiler	p.17
EDIT[OR]	Calls the EDITOR subsystem	p.17

File Manipulation Commands

COPY $\left\{ \begin{array}{l} \text{TEL[ETYPE]} \\ \text{TAPE} \\ \text{'filename}_1 \end{array} \right\}$ $\left\{ \begin{array}{l} \text{TO} \\ \text{OVER} \end{array} \right\}$ $\left\{ \begin{array}{l} \text{TEL[ETYPE]} \\ \text{TAPE} \\ \text{'filename}_2 \end{array} \right\}$	Creates, lists, and copies files	p.25
APPEND $\left\{ \begin{array}{l} \text{TEL[ETYPE]} \\ \text{TAPE} \\ \text{'filename}_1 \end{array} \right\}$ TO 'filename ₂ '	Adds a file to the end of another file	p.26
DELETE 'filename ₁ ' [, 'filename ₂ ', ... 'filename _i ']	Deletes one or more files from user's directory	p.27
RENAME 'filename ₁ ' AS 'filename ₂ '	Changes the name of a file	p.28
FILES ['filename ₁ '] [, filename ₂ ', ... 'filename _i ']	Produces a list of the file names and type of each from the user's file directory	p.29
DIRECTORY ['filename ₁ '] [, 'filename ₂ ', ... 'filename _i ']	Provides complete status information on files in the user's directory	p.30

File Protection Commands

LOCK 'filename'	Enables the user to define/alter the read/write passkeys and protection for a file	p.31
KEYS	Allows the user to enter passkeys for shared files	p.33

TENET USERS MANUAL EXECUTIVE

AUGUST 1970

**TENET
927 Thompson Place
Sunnyvale, California 94086**

Pub. No. 2002

Price \$3.00

PREFACE

This document is a user's manual which explains in general the capabilities of the TENET 210 Timesharing System, and in detail, the features, vocabulary, and usage of the EXECUTIVE. For more information about the subsystems discussed in this manual, refer to the following TENET documents:

BASIC USERS MANUAL
FORTRAN IV USERS MANUAL
EDITOR USERS MANUAL

FIRST EDITION:

Specifications contained herein are subject to changes which will be reported in subsequent revisions. Copies of this and other TENET publications can be obtained through TENET branch offices. A form is provided at the back of this publication for readers' comments. If the form has been detached, please direct your comments to TENET, 927 Thompson Place, Sunnyvale, California 94086 (Attention: Software Publications Department).

CONTENTS

PREFACE	iii	5 FILES	19
1 INTRODUCTION	1	Introduction	19
The TENET 210 Timesharing System	1	Data Type	19
EXECUTIVE	1	Record Type	20
TENET BASIC	3	Fixed-Length Records	20
FORTRAN IV	5	Variable-Length Records	20
EDITOR	7	File Names	21
Conventions Used in this Manual	8	Private Files	21
2 USER - SYSTEM COMMUNICATION	9	Shared Files	21
Connecting to the Computer	9	File Protection	21
Mode of Operation	9	File Access	22
Input Conventions	9	Shared File Names	23
Special Keys	9	Simultaneous Access to Files	23
Blanks	11	6 EXECUTIVE FILE COMMANDS	25
Prompt Characters	11	File Manipulation	25
3 LOGIN/LOGOUT PROCEDURE	13	COPY Command	25
LOGIN	13	APPEND Command	26
Passwords	13	DELETE Command	27
Correcting LOGIN Entries	13	RENAME Command	28
Sample LOGIN Sequences	14	FILES Command	29
Message of the Day	14	DIRECTORY Command	30
LOGOUT	14	File Protection	31
4 SUBSYSTEM CALLS	17	LOCK Command	31
BASIC	17	KEYS Command	33
FORTRAN	17	APPENDIX A ANSI Character Set	35
EDITOR	17	APPENDIX B Messages	37
Returning to EXECUTIVE	18	APPENDIX C Model 33 Teletypewriter Terminal	41
		INDEX	45

1. INTRODUCTION

THE TENET 210 TIMESHARING SYSTEM

The TENET 210 Timesharing System consists of a TENET 210 computer system and a set of interactive terminals that are connected to the computer by full duplex communications lines. The services provided by this system have been specifically designed for remote terminal users engaged in the on-line solution of a wide variety of problems in such diverse applications areas as business, science, engineering, and education. These services include:

- Highly interactive language processors such as TENET BASIC and FORTRAN IV
- Maintenance functions required by remote, on-line users involved in the creation, modification, and use of program and data files
- Media conversion facilities for transferring information between mass storage files and the system's peripheral devices (card reader, line printer, magnetic tape)

These services are provided by a large set of interdependent programs. Most of these programs that constitute the TENET 210 software system are not directly available to the user. Several, however, may be directly controlled by the user; these are the EXECUTIVE, the language processors, BASIC and FORTRAN IV, and the text editor, EDITOR.

EXECUTIVE

The EXECUTIVE is in essence the system administrator — it allows the user to access the system and subsequently, to access and move between available subsystems. EXECUTIVE provides the following services:

- LOGIN and LOGOUT modes

The user's first and last contact with the TENET Timesharing System is at the EXECUTIVE level. LOGIN and LOGOUT are procedures which govern user identification and accounting facilities. When a user first contacts the system, EXECUTIVE is in the

LOGIN mode, and the user is requested to identify himself. If there are any restrictions on the usage of a particular identification code, an appropriate password must also be given.

LOGOUT mode is also a function of the EXECUTIVE. When a user has finished work at the terminal he must return to EXECUTIVE to LOGOUT. He is then notified of accounting statistics such as the amount of Central Processor time used, terminal time, etc.

- File Maintenance

Operations which change and/or interrogate the status of user files may be performed at the EXECUTIVE level only. EXECUTIVE may be regarded as a catalog of information regarding all files currently maintained by the system, regardless of whether they were created at the EXECUTIVE level, or the subsystem level.

The user may also create files at the EXECUTIVE level to be used as data files for TENET BASIC, FORTRAN IV or EDITOR, or as program files for TENET BASIC or FORTRAN IV.

- File Access Control

All restrictions on file access and usage may be defined and satisfied at the EXECUTIVE level only. By designating passkey and protection values a user may control any subsequent usage of his file; similarly, only the file creator can change these values.

In order to access another user's file, the appropriate passkeys must be given at the EXECUTIVE level even though the file will be used at a subsystem level.

- Subsystem Access

EXECUTIVE control precedes and follows all subsystem activities. A set of EXECUTIVE commands enable the user to access any of the available subsystems. Once the user has completed work in one particular subsystem, he cannot directly move to another subsystem — he must first return to the EXECUTIVE level before he can move to another subsystem.

The functions and command vocabulary of EXECUTIVE are the subject of the remaining sections of this manual.

BASIC

BASIC is a high-level language processor originally designed to enable non-programmers to communicate with computers. Because the BASIC programming language is easy to learn, simple in content, and flexible in structure, it has become the language of timesharing systems. TENET BASIC is a greatly extended version of the original BASIC which incorporates many new features for both business and scientific applications.

- Four-Character Identifiers

As well as the letter-digit type identifier, the TENET BASIC programmer may use meaningful words to identify program variables. The program listing itself becomes a valuable documentation source.

- Variable Data-Type Declaration

For increased speed and precision, the user may optionally declare the data type of program variables — integer, real, double, complex, double complex, string.

- Matrix Operations

A complete matrix package is provided which allows matrix arithmetic, input/output, transposition and inversion.

- String Manipulation

TENET BASIC provides a complete string capability including concatenation, compression, and substring operations.

- Multi-Line Functions

The user may define functions of multiple as well as single lines. Each function may contain variables which are local to the function, that is, variables of the same name which appear in other functions or the main program represent different data. This distinction of variable scope enables the user to add a function to any program without jeopardizing the main program's data.

- Formatted Input/Output

Special input and output formats for both teletype and disc files can be easily specified and varied for report generation.

- Programmer-Defined Sequence Control

The simplest way to direct the computer to execute a statement out of sequence is to tell it to do that statement. At any point in a program, the programmer can specify selected statements to be executed using the DO statement. Once the selected statements are executed, program control is automatically returned to the statement following the DO statement.

- Function Library

TENET BASIC provides an extensive selection of functions: general mathematical, logarithmic, trigonometric, complex, string, hyperbolic, random number generation, tab control, and input/output.

- File Access

Data can be stored on disc in symbolic or binary form and in variable or fixed-length records. Access can be sequential or random. The user can have up to eight disc files active at one time, and each file can contain up to eight million characters.

- Program Linking

The program linking feature provides the capability for linking several related programs into one large program which essentially enables programs of unlimited size.

- Character Transmission Checking

The TENET Timesharing System receives information from the terminal in full duplex mode only. Each character is received by the computer and then echoed at the terminal. The full duplex capability documents exactly what the computer receives, thus minimizing undetected transmission errors.

- Syntax Error Checking

After each statement is entered from the terminal, it is checked for syntax errors. If any are found, the programmer is notified immediately, not after he attempts to execute his program. This feature can be deactivated at individual terminals, and syntax error checking deferred until the program is completed.

- Built-in Editing Facility

The TENET BASIC editing facility enables the user to review, rearrange, modify, and/or expand a program as and after it is generated within the BASIC subsystem.

- Interactive Environment

TENET BASIC is a truly interactive timesharing system. Interactive means that the user can control his program during execution as well as during program preparation. In an interactive environment, the same language that is used to write the program is used to control the actual program processing. The user is just as much a controlling agent as the system which processes the program. At any time, he can interrupt the normal sequence of execution, examine and/or alter the status of the program's data, create independent variables to be used on a temporary basis, and resume program execution at any point in the program. Within this context, TENET BASIC is not only a language to describe a program but a system to facilitate the interaction required for dynamic problem solution.

FORTRAN IV

TENET FORTRAN IV is an interactive language with special features which simplify writing syntactically correct programs and debugging. It is a superset of ANSI FORTRAN and compatible with IBM[†] FORTRAN IV-H with the following restrictions:

- Only disc and teletype input/output are allowed
- No ENTRY statement
- No EXTERNAL statement
- No IMPLICIT statement
- No RETURN option

TENET FORTRAN offers the following special features and options:

- **Multi-line Functions**
The user may define functions of multiple as well as single lines. Each function may contain variables which are local to the function, that is, variables of the same name which appear in other functions or the main program represent different data. This distinction of variable scope enables the user to add a function to any program without jeopardizing the main program's data.
- **Formatted Input/Output**
Special input and output formats for both teletype and disc files can be easily specified for report generation.
- **Free Form Input/Output**
The ACCEPT and DISPLAY commands enable the user to input and output information without format specification. Information is input and output in a default format whereby information fields are separated by commas.
- **NAMELIST**
The NAMELIST command enables the user to facilitate answering input requests at execution time by causing the name of the variable to be printed as a prompt for the appropriate value.
- **Direct Access Statements**
TENET FORTRAN IV direct access statements enable the user to define an exact location within a set of data for input and output operations.

[†] A registered trademark of International Business Machines Corporation

- END and ERR parameters are permitted in READ statements.
- T and Z format codes are allowed in format statements.
- Large Function Library

TENET FORTRAN IV provides an extensive selection of general functions: mathematical, logarithmic, trigonometric, complex, string, hyperbolic, random number generation, and input/output.

- Character Transmission Checking

The TENET Timesharing System receives information from the terminal in full duplex mode only. Each character is received by the computer and then echoed at the teletypewriter terminal. The full duplex capability documents exactly what the computer receives, thus minimizing undetected transmission errors.

- File Access

Data can be stored on disc in symbolic or binary form and in variable or fixed-length records. Access can be random or sequential. The user can have up to eight disc files active at one time, and each file can contain up to eight million characters.

- Program Linking

The program linking feature provides the capability for linking several related programs into one large program which essentially enables programs of unlimited size.

- Multi-Dimensioned Arrays

Arrays can be defined with a virtually unlimited number of dimensions.

- Mixed-Mode Expressions

The user may write expressions which consist of constants and variables of different as well as the same data types.

- Full Expression Capability in Subscripts

Expressions of any complexity may be used to define subscript values.

- Double exponentiation is permitted in expressions.
- Hexadecimal constants may be used as data initialization values.
- Initial data values may be assigned to variables in type statements.
- Literal constants may be enclosed in dollar signs as well as apostrophes.
- Syntax Error Checking

After each statement is entered from the terminal, it is checked for syntax errors. If any are found, the user is notified immediately, not after he attempts to execute his program. This feature can be deactivated at individual terminals, and syntax error checking deferred until the program is completed.

- **Built-in Editing Facility**

The FORTRAN IV editing facility enables the user to review, rearrange, modify and/or expand a program as and after it is generated within the FORTRAN IV subsystem.

- **Interactive Environment**

TENET FORTRAN IV is a truly interactive timesharing system. Interactive means that the user can control his program during execution as well as during program preparation. In an interactive environment, the same language that is used to write the program is used to control the actual program processing. The user is just as much a controlling agent as the system which processes the program. At any time he can interrupt the normal sequence of execution, examine and/or alter the status of the program's data, and create independent variables to be used on a temporary basis. Within this context, TENET FORTRAN IV is not only a language to describe a program but a system to facilitate the interaction required for dynamic problem solution.

EDITOR

The EDITOR subsystem is a sophisticated editing package which enables the user to manipulate the content of files to a greater degree than the built-in editing facilities of the TENET BASIC and FORTRAN IV subsystems. EDITOR consists of a set of programs which act on all files input to it as data, whether the files contain programs, data, or text. EDITOR offers the following features:

- **Modification by Line Number Reference**

EDITOR automatically prefaces every record of information on an input file with a line number that may be used in any EDITOR command to reference any record or range of records within a file.

- **Modification by Content Reference**

Instead of referencing records by line numbers, the user may specify a sequence of characters within a record in an EDITOR command.

- **Lines or ranges of lines may be moved, copied, deleted, inserted, or replaced.**

- **Characters within lines or ranges of lines may be deleted, replaced, or inserted.**

- **File Concatenation and Segmentation**

Files may be linked together with appropriate line renumbering, or individual files may be separated into multiple files.

- **Line Length Monitoring**

The user may keep all lines of a file within a prescribed length by using the LENGTH statement which warns the user when he exceeds the specified line length limit.

- Tab Setting

The user may specify up to four tab stops to be set on the terminal.

- Line Renumbering

The line numbers of a file may be resequenced, for example, after the file has been modified extensively.

CONVENTIONS USED IN THIS MANUAL

The following conventions are used throughout this manual:

- Upper case letters, digits, and special characters must appear exactly as shown in the format representation for all statements.
- Information in lower case letters in the format representations is to be supplied by the user.
- Braces { } indicate that one of the items enclosed must be used.
- Brackets [] indicate that the item (or items) is optional.
- Control characters are followed by the superscript ^c.
- Examples following the discussion of each statement are for the purpose of demonstrating the usage of the statements and are not necessarily examples of good programming practice.
- Underlined text in the examples indicates computer output or computer requests.

2. USER-SYSTEM COMMUNICATION

CONNECTING TO THE COMPUTER

The TENET Timesharing System receives and transmits information through a Teletypewriter Terminal. The method of connecting the terminal to the computer varies according to the type of the terminal (direct or acoustically coupled). (For information about the Model 33 Teletypewriter terminal and how to connect it to the computer, see appendix C.) Connection to the computer is confirmed by the system printing a message such as:

```
-----  
TENET TIMESHARING      6/7/70  
-----
```

The content of the header message is determined by individual installations.

INPUT CONVENTIONS

The terminal operates in the full duplex mode only. Characters input from the terminal are received by the computer and then typed out at the terminal; this process is referred to as echoing. The full duplex capability documents exactly what the computer receives, thus minimizing undetected transmission errors. Certain control characters are not echoed by the computer and words denoting special permissions are never echoed as a security precaution.

INPUT CONVENTIONS

Special Keys

The following special keys are used to transmit and modify information entered from the terminal. They can be used at any time input must be supplied by the user.

Ⓞ

The Carriage Return key signals an end of record. The teletypewriter print head is positioned at the beginning of the line. The system automatically supplies a Line Feed for every Carriage Return.

LF

The Line Feed key is used to enter records consisting of more than one line of information. Each time the Line Feed key is pressed, the paper is advanced one line. The system automatically supplies a Carriage Return for every Line Feed; this does not act as an end-of-record signal. The last line of a multi-line record must be terminated by a Carriage Return supplied by the user.

AC

The A control key deletes the previous character entered by the user. It echoes a backspace arrow (←) and eliminates a character each time the key is pressed: For example:

TENT AC ET 222 AC AC 10

is printed at the terminal as:

TENT ← ET 222 ←← 10

The system will interpret it as:

TENET 210

QC

The Q control key deletes the line currently being entered (i.e., before a CR is issued by the user). It echoes an upward arrow (↑) and a Carriage Return and Line Feed are automatically generated to position the teletype print head to the beginning of a new line. The user can then enter a new or corrected line.

If the QC key is used while entering multi-line statements, only the text following the most recent line feed will be deleted.

ESC or

ALT MODE

The Escape or ALT MODE key causes an interrupt of current operations. It causes all input/output operations at the terminal to be terminated; input not yet processed is lost. As this key has a different significance in other situations (e.g., during the LOGIN sequence), this key should not be used in place of the Q control key. Other uses of ESC are discussed where appropriate in this manual.

EOT

The EOT key terminates an input line and signals an end of record. However, unlike the Carriage Return or Line Feed keys, EOT does not cause the print head to be repositioned.

BREAK

On telephone coupled terminals the Break key causes a transmission interrupt and disconnects the terminal from the computer. Pressing this key could cause a loss of the current program and data. On direct-coupled terminals, this key is ignored.

BELL

The bell sounds whenever unrecognizable characters are entered and at line position 61 to warn the user that the 72-character teletypewriter line limit is approaching.

RUBOUT This key is used to delete characters on paper tape; it is ignored but does not ring the bell.

Blanks

It is necessary to separate EXECUTIVE command words, file names, constants, and variable names with one or more blanks. In general a blank is necessary after any item which could have the first character of the next item as a potentially valid character. Blank spaces may not be embedded in any variable name, file name, number, or operator.

Prompt Characters

All information entered from the terminal must be in response to a prompt signal from the system which indicates readiness to accept input. The following are the prompt characters used with the TENET timesharing systems:

Prompt Character	Systems	Input
-	EXECUTIVE	Any EXECUTIVE command
>	BASIC FORTRAN IV	Any BASIC command Any FORTRAN IV command
@	EDITOR	Any EDITOR command
?	All	Additional or corrective information In BASIC and FORTRAN IV, data for the program in execution.

3. LOGIN/LOGOUT PROCEDURE

LOGIN

Once the terminal is connected to the computer, it is automatically placed in the LOGIN command mode, and the user is requested to identify himself. In the following LOGIN sequence, the underlined text indicates system requests for information from the user.

-LOGIN account; name (CR)

where:

account = user's account number (0-511). Account number must be followed by a semicolon.

name = user's name (1-8 alphanumeric characters, % and \$, the first of which must be alphabetic, % or \$).

If either or both of the items requested by the LOGIN prompt are omitted or invalid, the user is reprompted by ACCOUNT? and/or NAME?.

Passwords

If the user has designated a password to be associated with his user name, the LOGIN sequence will include a prompt for a password (after the user enters account number; name (CR)).

As a security precaution password entries are not echoed at the teletypewriter.

Correcting LOGIN Entries

The control characters (A^c) and (Q^c) may be used during the LOGIN sequence to correct responses prior to pressing the Carriage Return key.

However, the (ESC) key in LOGIN mode causes the system to reissue the LOGIN prompt, thus requiring the user to start the LOGIN sequence again. The system allows the user three attempts to identify himself. After three unsuccessful attempts, or three minutes, the terminal is automatically disconnected.

Sample LOGIN Sequences

System Prompt	User Response	Echoed Text
-LOGIN PASSWORD?	115 (A ^c) 6; SMITH (CR) ADAM (CR)	-LOGIN 115 ← 6; SMITH PASSWORD?
-LOGIN ACCOUNT? NAME? PASSWORD?	(CR) 116 (CR) SMITH (CR) ADAM (CR)	-LOGIN ACCOUNT? 116 NAME? SMITH PASSWORD?
-LOGIN PASSWORD? -LOGIN PASSWORD?	116; SMITH (CR) ADAM (ESC) 116; SMYTH (CR) ADAM (CR)	-LOGIN 116; SMITH PASSWORD? -LOGIN 116; SMYTH PASSWORD?

Message of the Day

After the user has successfully completed the LOGIN sequence, a message of the day may be printed at the terminal. This message is automatically issued by the system before the user issues any EXECUTIVE level commands. Its content is variable and determined by individual installations. For example:

```
TENET TIME SHARING      8/18/70
```

LOGOUT

After the user completes his work at the terminal, he may leave the system by issuing the EXECUTIVE command LOGOUT. This command causes the following accounting information to be printed at the terminal:

```
ttt                      mm/dd/yy
CPU MINS - xx.xx
TERMINAL MINS - xx.xx
```

where:

```
ttt          = time in 24-hour clock units
mm/dd/yy    = month/day/year
xx.xx       = minutes
```

After this message, the system returns to LOGIN mode and prompts with "-LOGIN". If there is no response from the user within three minutes, or if the user turns the LINE/OFF/LOCAL knob to the OFF position, the terminal is disconnected.

4. SUBSYSTEM CALLS

One of the EXECUTIVE's functions is to provide access to any of the subsystems available to the user. For each of the subsystems, there is an EXECUTIVE level command consisting of the name of the subsystem. (All subsystem call commands can be abbreviated to their first four characters.) The appropriate subsystem call may be entered at any time in response to the EXECUTIVE prompt character (-). It is followed immediately by a Carriage Return. Once the subsystem is accessed and in control, the appropriate prompt character will be printed at the terminal.

BASIC

The EXECUTIVE command BASIC calls the BASIC compiler.
BASIC responds with the prompt character >.

```
-BASIC (CR)  
>
```

FORTRAN

The EXECUTIVE command FORTRAN calls the FORTRAN compiler.
FORTRAN responds with the prompt character >.

```
-FORT (CR)  
>
```

EDITOR

The EXECUTIVE command EDITOR calls the text editing package.
EDIT responds with the prompt character @.

```
-EDIT (CR)  
@
```

RETURNING TO EXECUTIVE

Once the user has completed work in one subsystem, he may transfer to the EXECUTIVE level by issuing the QUIT command. (QUIT is in the command vocabulary of each subsystem.)

He may then perform some operations at the EXECUTIVE level, call another subsystem, or terminate his session at the terminal by issuing the EXECUTIVE command LOGOUT (see Section 3, LOGOUT SEQUENCE).

5. FILES

INTRODUCTION

The TENET Timesharing System provides the user with a flexible means of creating, protecting, maintaining, and using files. A file is composed of structured information sets, called records, maintained on a mass storage device external to the computer's central memory.

The TENET system offers the user considerable latitude in designing files that best suit his needs. When a file is created the user determines its physical characteristics, usage within subsystems, and degree of privacy and protection.

Files are classified first as to data (content) type and record type. File data may be text or binary information; record type may be fixed-length or variable-length.

DATA TYPE

Information is stored on a file in text or binary form. Text files can be data or source (symbolic) language programs. The primary difference between data files and program files is one of usage — program files are primarily used by a language compiler such as BASIC or FORTRAN IV, while data files are used as input to the programs that these compilers produce. Text files correspond very closely to teletypewriter input/output. Text (including the carriage return and line feed) is maintained internally as four characters per word.

Binary files are written in internal machine language format (i. e., the translated version of information written in text form). Binary files store information more compactly and efficiently than text files. For example, the value $-0.123456E+6$ requires three words of internal storage in text format, but only one word in binary format. In addition, the time necessary for character-by-character processing required to convert to and from internal notation is eliminated.

Like text files, binary files can also be used to save programs; FORTRAN IV and BASIC both permit the binary form (object code) of the program to be saved, thus eliminating the need for compilation each time the program is subsequently executed.

RECORD TYPE

Every file has at least one and a maximum of 1,000,000 records. As records are written onto a file, they are identified not by exact position, but by a record number which serves as an index to the content of the file. By specifying the record's number, the user can readily access any record, or subset of information within a file. Record content may be altered without affecting the rest of the file as long as the replacement does not exceed the original in length.

Records may be of fixed or variable length. Fixed-length records allow rapid access to any part of a file since the exact location of the individual record can be determined directly. Variable-length records are not restricted to a particular length; as such, their exact position cannot be determined directly. Thus, they are somewhat slower to access than fixed-length records.

Fixed-Length Records

The user must specify the length (up to 4096 bytes) of each record of a fixed-length record file when the file is created. Since each record in a fixed-length record file is the same size, the entire file need not be created in sequence. Thus, the user may create record three before records one and two. Space is automatically reserved for implicitly defined (non-written) records.

Since disc space is allocated for all records up to the record number specified, an attempt to read an unwritten record in the specified range will not generate an error condition. The content of implicitly defined records are, however, unpredictable under these circumstances.

Fixed-length record files are generally used where access time is important. However, if the variance of actual record size within a fixed-length record file is considerable, storage efficiency will be compromised.

Variable-Length Records

A variable-length record can be as large as 8,000,000 bytes (as large as the largest file allowed). Unlike fixed-length record files, a file containing variable-length records must be written (created) in sequence since the user does not specify record length when the file is created. Each new record in a variable-length file must be assigned a record number one greater than the largest current number. Thus, the user could not write record three before records one and two. Implicitly defined (non-written) records are not possible in a variable-length record file.

Although variable-length files must be created sequentially, they can be accessed randomly. However, they cannot be referenced as quickly as fixed-length records, and are generally used for applications which access file content sequentially or which require low volume random processing.

FILE NAMES

A file is addressable by a unique name assigned by the user when the file is created. The name consists of a string of from one to eight alphanumeric characters, including % and \$. The first character must be alphabetic, % or \$. The following are legal file names: FILE2, %%RATE%%, \$COST\$, MYFILE.

PRIVATE FILES

All files created by a user are assumed to be private; i. e. , no other user may access the file for any purpose. A file can be made shareable only by the file creator's deliberate specification.

SHARED FILES

To make a file available to other users, the file's creator must: define the type of operations allowed on the file, set passkey values which must be used to gain access to the file, and make his own account number and user name known to those users sharing the file.

File Protection

The file's creator can protect his file by declaring it Read/Write or Read Only. Read/Write specification means that both operations (read/write) may be performed by anyone with legitimate access to the file. The file is essentially unprotected by this declaration. However, only the file's creator may delete the file.


A file declared as Read Only is protected from write operations by anyone sharing the file, including the file creator himself. The file cannot even be deleted by its creator while it has Read Only status; the file must first be unprotected (declared Read/Write) before it can be deleted.

File Access

Passkey values enable other users to gain access to a shared file. A passkey value is associated with each operation allowed on the file. Thus, there are separate passkey values set for reading and writing a shared file. The creator of the file is not required to specify passkey values when accessing a shared file.

<u>Read or Write Passkey Value</u>	<u>Meaning</u>
0000	Not shared. The file may not be read (written) by anyone but the file's creator.
FFFF	Shared. The file may be read (written) by appending the file creator's account number and name to the name of the file.
0001 - FFFE	Shared. The file may be read (written) by anyone specifying the correct passkey values and appending the file creator's account number and name to the name of the file.

Protection declarations and passkey values allow the user a wide variety of different types of restrictions on access to his files. The range of protections a user may specify for a file are as follows:

	<u>File</u>	<u>Protection</u>	<u>Read Passkey Value</u>	<u>Write Passkey Value</u>
most protected	1	Read Only	0000	0000
	2	Read/Write	0000	0000
	3	Read Only	0001-FFFE	0000
	4	Read Only	FFFF	0000
	5	Read Only	FFFF	0001-FFFE
	6	Read/Write	0001-FFFE	0001-FFFE
	7	Read/Write	FFFF	FFFF
	least protected			

File 1 is most protected. It is Read Only and can be accessed by its creator only. File 2 is a private file and may be read or written by its creator only. File 3 is Read Only and can be accessed only by users who specify the correct passkey value and append the account number and creator name to the name of the file. Files 4 and 5 are equivalent, as the Read Only protection overrides a passkey value for writing.

Shared File Names

To access a shared file, a user must specify the account number and name of the file's creator (in addition to specifying passkey values where required). When a user actually opens a file for an operation, he must preface this information to the name of the file as follows:

file name = a;un;fn

where:

a = creator's account number (0-511)

un = creator's user name (1-8 alphanumeric characters, % and \$, the first of which must be alphabetic, % or \$)

fn = file name (1-8 alphanumeric characters, % and \$, the first of which must be alphabetic, % or \$)

This form of a shared file's name is required only for users other than the file's creator.

SIMULTANEOUS ACCESS TO SHARED FILES

A shared file may be read simultaneously by any number of users. However, it may not be written (modified) by more than one user at a time. The following rules govern simultaneous access to shared files:

- A user may successfully access a file for reading (as though he were the only one using the file) while the file is being read by other users.
- An attempt to read a file currently being written will be rejected.
- An attempt to write a file currently being read or written by another user will be rejected.

The above rules pertain to the file creator as well as other users.

6. EXECUTIVE FILE COMMANDS

FILE MANIPULATION

COPY Command

$\text{COPY} \left\{ \begin{array}{l} \text{TEL[ETYPE]} \\ \text{TAPE} \\ \text{'file name}_1\text{' } \end{array} \right\} \left\{ \begin{array}{l} \text{TO} \\ \text{OVER} \end{array} \right\} \left\{ \begin{array}{l} \text{TEL[ETYPE]} \\ \text{TAPE} \\ \text{'file name}_2\text{' } \end{array} \right\}$
--

The EXECUTIVE command COPY enables the user to create files from the terminal, list files at the terminal, and copy one file to another.

- TO should be used when the destination file (filename₂) is new; OVER should be used when the destination file already exists.
- User file names (private or shared) must be enclosed in single quotation marks.
- The file name TELETYPE (or TEL) must be used for input or output to the teletypewriter. It should not be enclosed in quotation marks.
- The file name TAPE must be used for file input and output to the paper tape device. It should not be enclosed in quotation marks.
- Teletype input is terminated when a null record ending with **EOT** is entered from the terminal (from the teletypewriter or paper tape). This record is not entered into the file.

Example:

```
-COPY TELETYPE TO 'FILE1' CR
ADAMS, ROGER P., 104-34-3670, MISSOURI CR
CALLER, SHIRLEY R., 114-90-4213, CALIFORNIA CR
DELLMAN, STANLEY J., 90-43-6219, OREGON CR
DEWITT, ELAINE, 124-67-3290, NEVADA CR
:
:
- EOT
-COPY 'FILE1' TO 'SOCSEC' CR
```

In the first entry the user notifies the system that he will be creating the new file, 'FILE1', from the teletype keyboard. After entering multiple records, input to the file is terminated by pressing the **EOT** key. In the last command, 'FILE1' is copied to another new file, 'SOCSEC'. 'FILE1' is not deleted by this operation.

APPEND Command

$\text{APPEND} \left\{ \begin{array}{l} \text{TEL[ETYPE]} \\ \text{TAPE} \\ \text{'file name}_1\text{' } \end{array} \right\} \text{ TO 'file name}_2\text{'}$
--

The EXECUTIVE command APPEND adds a file to the end of an existing file.

- User file names (private or shared) must be enclosed in single quotation marks.
- The file name TELETYPE or TEL must be used for input from the teletypewriter. It should not be enclosed in quotation marks.
- The file name TAPE must be used for input from the paper tape device. It should not be enclosed in quotation marks.
- Input entered from the teletype terminates when a null record ending with **(EOT)** is entered from the terminal. This record is not appended to the file.

Example:

```
-COPY TELETYPE TO 'MYFILE' (CR)
10 READ A0, B0, C0 (CR)
20 DATA 2.3, 3.3, 4.3 (CR)
30 LET X0 = A0 * B0 (CR)
40 IF X0 > RATE GOTO 90 (CR)
50 X0 = X0 * C0 (CR)
60 DO 40 (CR)
70 PRINT "X0="; X0 (CR)
90 PRINT "END" (CR)
100 END (CR)
(EOT)
-APPEND TELETYPE TO 'MYFILE' (CR)
25 RATE = C0/B0 (CR)
(EOT)
```

The COPY command causes all subsequent lines (up to the **(EOT)**) to be copied from the teletype to the new file 'MYFILE'. The user then adds an additional statement to his BASIC program by using the APPEND command. The APPEND command causes subsequent input up to **(EOT)** to be added to the end of 'MYFILE'.

DELETE Command

```
DELETE 'file name1' [, 'file name2', ... 'file namei']
```

The EXECUTIVE command DELETE removes one or more files from the user's file directory and makes the disc space allocated to these files available for reassignment.

- The user file names must be enclosed in single quotation marks.
- 'file name_i' designates the file to be deleted and must be owned by the user. (Only the creator of a file may delete it.)

Example:

```
-COPY 'MYFILE' TO TAPE (CR)  
-DELETE 'MYFILE' (CR)
```

After the user causes the contents of the file 'MYFILE' to be printed out at the terminal, he deletes the original copy of the file from disc storage.

RENAME Command

```
RENAME 'file name1' AS 'file name2'
```

The EXECUTIVE command RENAME changes the name of a file created by the user.

- File names must be enclosed in single quotation marks.
- 'file name₁' designates the old file name.
- 'file name₂' designates the new file name.
- Only the file creator may alter the name of a file.

Example:

```
-RENAME 'ONE' AS 'TWO' (CR)
```

FILES Command

`FILES ['file name1'] [, 'file name2', ... 'file namei']`

The EXECUTIVE command FILES produces a list of the file names and the type of each from the user's file directory. File type is defined as follows:

T = text
B = binary
BB = BASIC binary
FB = FORTRAN binary

- All file names must be enclosed in single quotation marks.
- 'file name_i' designates the file which will be listed by name and type. The names of shared files not created by the user must be preceded by their associated account number and user name.
- If no parameters are specified after the FILES command, a list of all the files in the user's directory will be produced.

Examples:

```
-FILES (CR)
NAME      TYPE
-----
JONES1    T
MYFILE    B
HISFILE   T
HERFILE   T
JONES2    B
JONES3    BB
```

The FILES command causes the directory associated with the current user to be printed.

```
-FILES 'MYFILE', 'HISFILE', 'HERFILE' (CR)
NAME      TYPE
-----
MYFILE    B
HISFILE   T
HERFILE   T
```

The FILES command in this case is effectively a request for the type of the files specified.

DIRECTORY Command

```
DIRECTORY ['file name1'] [, 'file name2', ... 'file namei']
```

The EXECUTIVE command DIRECTORY is similar to the FILES command, but provides additional status information about the files. It generates the following information: file name, file type (T = text, B = binary, BB = BASIC binary, FB = FORTRAN binary), size in characters, date created, date last updated, and number of times accessed since date of creation.

- All file names must be enclosed in single quotation marks.
- 'file name₁' designates the file to be listed by name and description. The names of shared files not created by the user must be preceded by their associated account number and user name.
- If no parameters are specified after the DIRECTORY command, a list of all the files in the user's directory will be produced.

Example:

```
-DIRECTORY CR  
NAME    TYPE  SIZE  CREATED  UPDATED  REF  
SMITH   T       512    01/06/69  02/28/69  94  
SMITH2  BB      224    09/23/69  11/04/69  10  
SMITH3  T       106    12/31/69  01/16/70  5  
MAFILE  B       410    10/31/69  11/02/69  20
```

FILE PROTECTION

LOCK Command

LOCK 'file name'

The EXECUTIVE COMMAND LOCK enables the user to define or alter the read and write passkeys and protection on a file (created by the user).

- 'file name' is the name of the file whose protection will be defined or altered.
- No distinction is made between defining the protection of a file and altering the protection of a file.

Once the LOCK command is entered, the system responds with:

ENTER RPK, WPK, RW/RO!

The user should then enter the new passkeys (in hexadecimal) and protection code in the order specified by the system request.

RPK = read passkey value
WPK = write passkey value
RW/RO = read/write or read only protection

The file access and protection parameters are not echoed by the system. If a parameter is not specified (an entry consisting of only a comma), the current access or protection code is left unchanged. If a file's access or protection is being defined for the first time, and a parameter is not specified, its default value is used. The default value for both passkeys is 0000; the default for protection is RW.

Passkeys are defined as follows:

0000	Not shareable for reading (writing)
FFFF	Shared. Anyone specifying the correct account number/user name combination may access the file for reading (writing).
0001-FFFE	Shared. Anyone specifying the correct account number/user name combination and the passkey value may access the file for reading (writing).

Example:

```
-LOCK 'FILE07' (CR)  
ENTER RPK, WPK, RW/RO! FFFF,,RO (CR)
```

The read passkey is defined to make the file accessible to anyone; the write passkey is unchanged (or = 0000 if the first definition of the write passkey); the only operation allowed on the file is read.

KEYS Command

KEYS

The EXEC command KEYS enters a series of passkeys necessary to access shared files for a programming session at the terminal. Once the user has submitted the appropriate passkeys at the EXECUTIVE level, he may access the shared files at the EXECUTIVE or any subsystem level.

The system responds to the KEYS command with the following request:

ENTER KEYS!

The user should then enter the passkeys necessary for the files he wants to access. Passkey values must be entered in hexadecimal and are not echoed. They may be entered in any order and it is not necessary to enter duplicate passkey values for different files.

If the user enters an improper (i. e. , unrecognizable) passkey, the system will print out the error message

ENTRY xx INCORRECT PASSKEY

xx is the sequential number of the incorrect passkey; e. g. , ENTRY 04 specifies the fourth passkey entered. Following this message the system will repeat the ENTER KEYS! prompt. The user should then complete the passkey list starting with the passkey in error.

Example:

```
-KEYS (CR)  
ENTER KEYS! A EFF, ABCD, 1234, 0010, AA00 (CR)
```


APPENDIX A. ANSI CHARACTER SET

ANSI Hex Code	Character	Teletype Key	Teletype/Printer Graphic	Hollerith Card Code	ANSI Hex Code	Character	Teletype Key	Teletype/Printer Graphic	Hollerith Card Code
00	NUL	P ^{CS}		12-0-9-8-1	25	%	5 ^S	%	0-8-4
01	SOH or DEL	A ^C		12-9-1	26	&	6 ^S	&	12
02	STX	B ^C		12-9-2	27	'	7 ^S	'	8-5
03	ETX	C ^C		12-9-3	28	(8 ^S	(12-8-5
04	EOT	D ^C		9-7	29)	9 ^S)	11-8-5
05	ENQ	E ^C		0-9-8-5	2A	*	: ^S	*	11-8-4
06	ACK	F ^C		0-9-8-6	2B	+	; ^S	+	12-8-6
07	BEL	G ^C		0-9-8-7	2C	,	,	,	0-8-3
08	BS	H ^C		11-9-6	2D	-	-	-	11
09	HT	I ^C		12-9-5	2E	.	.	.	12-8-3
0A	LF	Line Feed		0-9-5	2F	/	/	/	0-1
0B	VT	K ^C		12-9-8-3	30	0	0	0	0
0C	FF	L ^C		12-9-8-4	31	1	1	1	1
0D	CR	Return		12-9-8-5	32	2	2	2	2
0E	SO	N ^C		12-9-8-6	33	3	3	3	3
0F	SI	O ^C		12-9-8-7	34	4	4	4	4
10	DLE	P ^C		12-11-9-8-1	35	5	5	5	5
11	DC1	Q ^C		11-9-1	36	6	6	6	6
12	DC2	R ^C		11-9-2	37	7	7	7	7
13	DC3	S ^C		11-9-3	38	8	8	8	8
14	DC4	T ^C		9-8-4	39	9	9	9	9
15	NAK	U ^C		9-8-5	3A	:	:	:	8-2
16	SYN	V ^C		9-2	3B	;	;	;	11-8-6
17	ETB	W ^C		0-9-6	3C	<	, ^S	<	12-8-4
18	CAN	X ^C		11-9-8	3D	=	- ^S	=	8-6
19	EM	Y ^C		11-9-8-1	3E	>	. ^S	>	0-8-6
1A	SUB	Z ^C		9-8-7	3F	?	/ ^S	?	0-8-7
1B	ESC	K ^{CS}		0-9-7	40	@	P ^S	@	8-4
1C	FS	L ^{CS}		11-9-8-4	41	A	A	A	12-1
1D	GS	M ^{CS}		11-9-8-5	42	B	B	B	12-2
1E	RS	N ^{CS}		11-9-8-6	43	C	C	C	12-3
1F	US	O ^{CS}		11-9-8-7	44	D	D	D	12-4
20	Blank	Space Bar			45	E	E	E	12-5
21	!	1 ^S	!	12-8-7	46	F	F	F	12-6
22	"	2 ^S	"	8-7	47	G	G	G	12-7
23	#	3 ^S	#	8-3	48	H	H	H	12-8
24	\$	4 ^S	\$	11-8-3	49	I	I	I	12-9
					4A	J	J	J	11-1

ANSI Hex Code	Character	Teletype Key	Teletype/Printer Graphic	Hollerith Card Code	ANSI Hex Code	Character	Teletype Key	Teletype/Printer Graphic	Hollerith Card Code
4B	K	K	K	11-2	66	f		F	12-0-6
4C	L	L	L	11-3	67	g		G	12-0-7
4D	M	M	M	11-4	68	h		H	12-0-8
4E	N	N	N	11-5	69	i		I	12-0-9
4F	O	O	O	11-6	6A	j		J	12-11-1
50	P	P	P	11-7	6B	k		K	12-11-2
51	Q	Q	Q	11-8	6C	l		L	12-11-3
52	R	R	R	11-9	6D	m		M	12-11-4
53	S	S	S	0-2	6E	n		N	12-11-5
54	T	T	T	0-3	6F	o		O	12-11-6
55	U	U	U	0-4	70	p		P	12-11-7
56	V	V	V	0-5	71	q		Q	12-11-8
57	W	W	W	0-6	72	r		R	12-11-9
58	X	X	X	0-7	73	s		S	11-0-2
59	Y	Y	Y	0-8	74	t		T	11-0-3
5A	Z	Z	Z	0-9	75	u		U	11-0-4
5B	[K ^S	[12-8-2	76	v		V	11-0-5
5C	\	L ^S	\	0-8-2	77	w		W	11-0-6
5D]	M ^S]	11-8-2	78	x		X	11-0-7
5E	†	N ^S	†	11-8-7	79	y		Y	11-0-8
5F	←	O ^S	←	0-8-5	7A	z		Z	11-0-9
60	↘ or ¢		@	8-1	7B	{			12-0
61	a		A	12-0-1	7C				12-11
62	b		B	12-0-2	7D	}			11-0
63	c		C	12-0-3	7E	┘ or ~			11-0-1
64	d		D	12-0-4	7F	RUBOUT			12-9-7
65	e		E	12-0-5					

APPENDIX B. MESSAGES

This appendix contains a complete list of all messages issued by the EXECUTIVE that request user input and that respond to user errors. In addition, a set of messages (not documented in this manual) are generated by the system itself. In most cases system errors cause job termination; the user should phone the computer operator for information.

In the following messages xx points to a parameter entered by the user. For example 04 refers to the fourth item (other than an EXECUTIVE command word) entered by the user. In messages pertaining to files xx = 01 for input and xx = 02 for output files.

ACCOUNT?

- Cause: 1. An account number was not entered during the LOGIN sequence.
2. An invalid account number was entered during the LOGIN sequence.

Action: Enter correct account number followed by a **CR**.

EH?

Cause: The user has replied to a system prompt with an invalid or unrecognizable word.

Action: Reenter correct word after system prompt.

ENTER KEYS!

Cause: System response to the KEYS command.

Action: Enter the passkeys necessary for the files to be accessed during the session at the terminal.

ENTER RPK, WPK, RW/RO!

Cause: System response to the LOCK command.

Action: Enter new passkeys (in hexadecimal) and protection code for file.

ENTRY xx DUPLICATE FILE NAME

Cause: An (output) file name specified in a COPY or RENAME command already exists in the user's directory.

- Action: 1. Delete the old file before creating (renaming) a new file of the same name.
2. Change the name of the new file.

ENTRY xx FILE NAME NOT DEFINED

Cause: The file name supplied for a file was in error.

Action: Reenter the command with the correct file name.

ENTRY xx FILE NOT SHARABLE

Cause: The file specified is private and inaccessible by the user.

Action: None

ENTRY xx INCORRECT PASSKEY

Cause: 1. An incorrect passkey was entered in response to the ENTER KEYS! prompt.
2. The file's creator has changed the passkey value for the file.

Action: Reenter passkey value list with correct passkey values beginning with entry xx.

ENTRY xx INVALID ACCOUNT NUMBER

Cause: The account number specified for a shared file is incorrect.

Action: Reenter the command with the correct account number.

ENTRY xx NOT A TEXT FILE

Cause: The file specified is in binary format and as such cannot be accessed from the EXECUTIVE level.

Action: None

ENTRY xx NOT A WRITE FILE

Cause: An attempt was made to write on a shared file designated as Read Only by its creator.

Action: None

ENTER xx USER'S NAME UNDEFINED

Cause: The user name entered for a shared file was in error.

Action: Reenter the command with the correct user name.

ENTRY xx SYNTAX ERROR

Cause: The format of the command as entered was in error.

Action: Reenter the command in the correct format.

ERROR

Cause: The account number, user name or password has been entered incorrectly during the LOGIN sequence. The user will be reprompted for the appropriate item.

Action: Reenter the correct account number, user name or password.

LOGIN

Cause: The terminal has been connected to the computer and is requesting the user to identify himself.

Action: Enter the account number and user name separated by a semi-colon and followed by a (CR). If there is a password associated with the user name, the system will print

PASSWORD?

The user should respond with the correct password followed by a (CR).

NAME ?

- Cause: 1. A user name was not entered during the LOGIN sequence.
2. An invalid user name was entered during the LOGIN sequence.
- Action: Enter the correct user name followed by a (CR).

NAME IN USE

- Cause: Another user is already using the account number and user name specified during the LOGIN sequence.
- Action: None. The user cannot currently access the system.

PASSWORD ?

- Cause: There is a password associated with the user name entered during the LOGIN sequence.
- Action: Enter the correct password followed by a (CR).

SHARED FILE BUSY

- Cause: An attempt to read (write) a shared file already in use by another user for a write (read/write) operation.
- Action: None. The file cannot be accessed currently.

USER DIRECTORY FULL

- Cause: An attempt has been made to create a file for which there is no room in the user's directory.
- Action: Delete old, obsolete files.

USER LIMIT EXCEEDED

- Cause: The user has exceeded the maximum central processor time, terminal time, or disc space associated with the account number and user name.
- Action: None. The user cannot currently access the system.

APPENDIX C. MODEL 33 TELETYPEWRITER TERMINAL

The Model 33 Teletypewriter Terminal used by the TENET Timesharing System consists of a control unit, keyboard, paper tape punch, and paper tape reader mechanism.

CONTROL UNIT

The configuration of the control unit (see Figure C-1) depends on whether the terminal is direct or acoustically coupled to the computer. The control unit on a direct-coupled terminal consists of only a LINE/OFF/LOCAL knob.

- LINE If the control is in the LINE position, the terminal should be on and connected to the computer (on-line) unless the terminal had been automatically disconnected. In this case turn the knob first to the OFF position, and then to the LINE position to establish a connection.
- OFF The terminal is off and incapable of communicating with the computer.
- LOCAL The terminal is on but not connected to the computer. When the terminal is in this mode (off-line), operations such as punching paper tape may be performed.

Acoustically coupled terminals require a headset mechanism which is used to hold a telephone receiver. The LINE/OFF/LOCAL knob for acoustically coupled terminals is the same as for direct-coupled terminals except that when the knob is in the LINE position, the terminal is not automatically connected to the computer, but is capable of being connected to the computer. To establish a connection, the user must first turn the knob to the LINE position, phone the computer site, wait for a high-pitched tone, and place the telephone receiver into the headset mechanism.

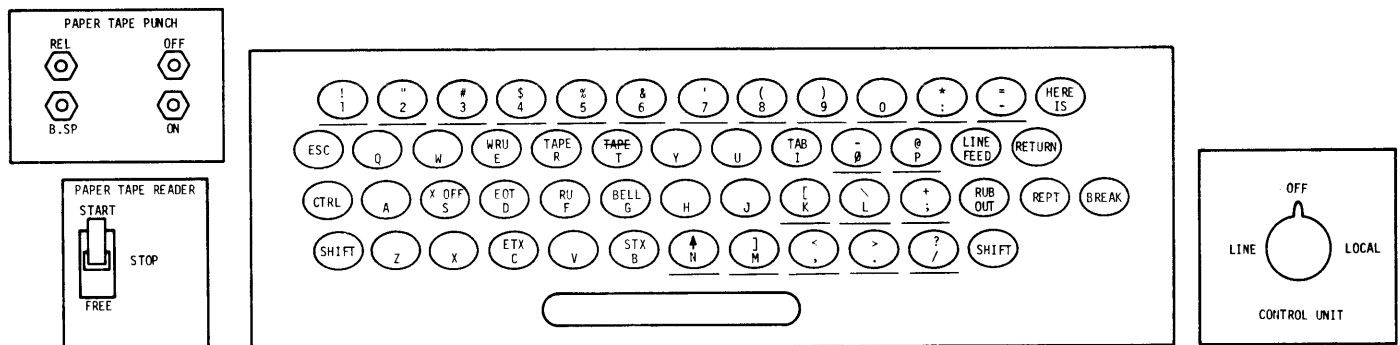


Figure C-1. Model 33 Teletypewriter Terminal Keyboard and Controls

THE KEYBOARD

The teletype keyboard (see Figure C-1) is used as a standard typewriter keyboard with the exception of the keys described below.

SHIFT	<p>Only those keys underlined in Figure C-1 have a shift position. The shift key is non-locking and must be depressed when typing. Characters are printed as they appear on the upper half of the key. However, on some terminals:</p> <p style="padding-left: 40px;">K shift is not marked but appears as a [</p> <p style="padding-left: 40px;">L shift is not marked but appears as a \</p> <p style="padding-left: 40px;">M shift is not marked but appears as a]</p> <p>The keyboard locks whenever an attempt is made to use the shift with a key with no shift position.</p>
CTRL (Control)	<p>Any alphabetic character may be pressed in conjunction with CTRL. (CTRL is non-locking.) The resulting control character is not always printed at the terminal. Characters used with the CTRL are discussed in section 9 of this manual. They are designated by the subscript ^c. Control characters not recognized by the system are ignored but cause the bell to ring once for each ignored character.</p>
ESC or ALT MODE	<p>This key terminates any input/output operation in progress and causes a program interrupt.</p>
LINE FEED	<p>Each time the Line Feed key is pressed, the paper is advanced one line. (When the terminal is connected to the computer, the system automatically generates a Carriage Return for each Line Feed.)</p>
RETURN or CARRIAGE RETURN	<p>This key positions the print head at the beginning of a line. When the terminal is connected to the computer (on-line), the system automatically generates a Line Feed for every Carriage Return.</p>
RUBOUT	<p>This key is used to delete characters on paper tape. It is always ignored, but does not ring the bell.</p>
REPT (Repeat)	<p>This key causes any character key pressed while the REPT is pressed to be repeated for as long as the REPT key is pressed.</p>
HERE IS	<p>Transmits and prints whatever is on the answerback drum.</p>
BREAK	<p>On a direct coupled terminal this key is ignored; on an acoustically coupled terminal, pressing this key disconnects the terminal from the computer.</p>

PAPER TAPE PUNCH

The paper tape punch is used to produce a perforated tape which can be used as input to the computer instead of input from the teletypewriter terminal keyboard.

OFF and ON	The ON button initiates and continues paper tape punching until the OFF button is pressed. Information punched on paper tape is also printed at the terminal.
REL	The release button frees the paper tape so that the user can manually pull blank tape through the punch mechanism.
BKSP	The backspace button moves the paper tape backwards one frame each time the button is pressed. It is used in conjunction with the RUBOUT key to delete paper tape entries.

PREPARING PAPER TAPE OFF-LINE

The user can save information in paper tape form. The TAPE command, for example, is used by the BASIC subsystem to access programs saved on paper tape. Similarly, the contents of data files may be punched on paper tape and later read into a file by the EXECUTIVE. To prepare paper tape off-line, turn the terminal control dial to LOCAL, depress the Punch ON button, and enter data from the keyboard. Since this is an off-line operation, the user will find it convenient to follow a Line Feed with a Carriage Return and vice versa. When paper tapes are read by the system, Line Feed / Carriage Return and Carriage Return / Line Feed combinations are treated as Line Feed and Carriage Return respectively.

The control keys \textcircled{A}^c (delete previous character) and \textcircled{Q}^c (delete current line) may be used to edit paper tape entries.

PAPER TAPE READER

START	This control initiates and continues paper tape reading.
STOP	This key terminates paper tape reading.
FREE	This control frees the reader mechanism so that the tape can be pulled through the reader manually.

INDEX

- A control key 10, 13
- Accessing shared files 22, 33
- ACCOUNT? 13
- Account numbers 13
- Accounting information 14
- Acoustically-coupled terminals 41
- ALT MODE key 10, 42
- ANSI character set 35
- APPEND command 26

- BASIC command 17
- Bell (Teletypewriter) 10
- Binary files 19
- Blanks 11
- BREAK key 10, 42

- Carriage return key 9, 42
- Connecting to the computer 9, 42
- Control key 42
- Control unit, terminal 40
- COPY command 25
- CTRL key 42

- Data files 19
- DELETE command 27
- Deleting characters 10, 11
- Deleting statements 10
- Direct-coupled terminals 41
- DIRECTORY command 30
- Disconnecting the terminal 13, 15

- Echoing 9
- EDITOR command 17
- EOT key 10, 25
- Error messages 37
- ESC key 10, 13, 42

- EXECUTIVE 1
 - returning to 18
- EXECUTIVE commands
 - APPEND 26
 - BASIC 17
 - COPY 25
 - DELETE 27
 - DIRECTORY 30
 - EDITOR 17
 - FILES 29
 - FORTRAN 17
 - KEYS 33
 - LOCK 31
 - LOGOUT 14
 - RENAME 28
- Extending files 26

- Files
 - access 20
 - binary 19
 - commands 22
 - creation 25
 - deleting 27
 - directory 30
 - fixed-length record 20
 - manipulation commands 25
 - names 19
 - names, shared 23
 - private 21
 - protection 21, 22
 - protection commands 31
 - renaming 28
 - shared 21
 - simultaneous access to 21
 - status 30
 - text 19
 - variable-length record 20
- FILES command 29
- Fixed-length records 20
- FORTRAN command 14, 17
- FORTRAN IV 5
- Full duplex mode 9

- Header message 9
- HERE IS key 42
- Hexadecimal codes 35

Keyboard, terminal 41, 42
KEYS command 33

Line feed key 10, 42
LOCK command 31
LOGIN sequence 13
LOGIN sequence samples 14
LOGOUT command 14
LOGOUT sequence 14

Message of the day 14
Messages 37
Model 33 Teletypewriter Terminal 41
Multi-line statements 10

NAME? 13

Off-line operation 43

Paper tape files 25
Paper tape preparation 43
Paper tape punch 43
Paper tape reader 43
Passkeys 22, 31
Passwords 13
Preparing paper tape off-line 43
Private files 21
Program files 19
Prompt characters 11
Prompts, LOGIN 13
Protecting files 21
Protections, file 22

Q control key 10, 13
QUIT command 18

Read Only files 21
Read/Write files 21

Records
 fixed-length 20
 implicitly defined 20
 size 20
 variable-length 20
RENAME command 28
REPEAT key 42
REPT key 42
RETURN key 42
Return to EXECUTIVE 18
RUBOUT key 11, 42

Security, user name 13
Shared file names 23
Shared files 21
Shift key 42
Signing on to the system 13
Simultaneous file access 23
Special keys
 A^c 10
 ALT MODE 10
 BREAK 10
 CR 9
 EOT 10
 ESC 10
 LF 10
 Q^c 10
 RUBOUT 11
Subsystem calls 17

Teletype files 25
Teletypewriter terminal 41
TENET BASIC 3
TENET 210 Timesharing System 1
Terminal 41
Text files 19

User identification 13
User name 13

Variable-length records 20

FOLD

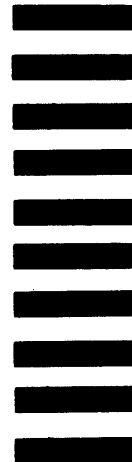
FIRST CLASS
PERMIT NO. 480
SUNNYVALE, CA.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A

POSTAGE WILL BE PAID BY

TENET
927 THOMPSON PLACE
SUNNYVALE, CA. 94086

ATTENTION: SOFTWARE PUBLICATIONS



FOLD

FROM: NAME _____

POSITION _____

ADDRESS _____

EXECUTIVE COMMAND SET

Subsystem Call Commands

BASI[C]	Calls the TENET BASIC compiler	p.17
FORT[RAN]	Calls the FORTRAN IV compiler	p.17
EDIT[OR]	Calls the EDITOR subsystem	p.17

File Manipulation Commands

COPY $\left\{ \begin{array}{l} \text{TEL[ETYPE]} \\ \text{TAPE} \\ \text{'filename}_1 \end{array} \right\}$ $\left\{ \begin{array}{l} \text{TO} \\ \text{OVER} \end{array} \right\}$ $\left\{ \begin{array}{l} \text{TEL[ETYPE]} \\ \text{TAPE} \\ \text{'filename}_2 \end{array} \right\}$	Creates, lists, and copies files	p.25
APPEND $\left\{ \begin{array}{l} \text{TEL[ETYPE]} \\ \text{TAPE} \\ \text{'filename}_1 \end{array} \right\}$ TO 'filename ₂ '	Adds a file to the end of another file	p.26
DELETE 'filename ₁ ' [, 'filename ₂ ', ... 'filename _i ']	Deletes one or more files from user's directory	p.27
RENAME 'filename ₁ ' AS 'filename ₂ '	Changes the name of a file	p.28
FILES ['filename ₁ '] [, filename ₂ ', ... 'filename _i ']	Produces a list of the file names and type of each from the user's file directory	p.29
DIRECTORY ['filename ₁ '] [, 'filename ₂ ', ... 'filename _i ']	Provides complete status information on files in the user's directory	p.30

File Protection Commands

LOCK 'filename'	Enables the user to define/alter the read/write passkeys and protection for a file	p.31
KEYS	Allows the user to enter passkeys for shared files	p.33

TENET, Inc. / 927 Thompson Place / Sunnyvale, California 94086 / (408) 245-8751