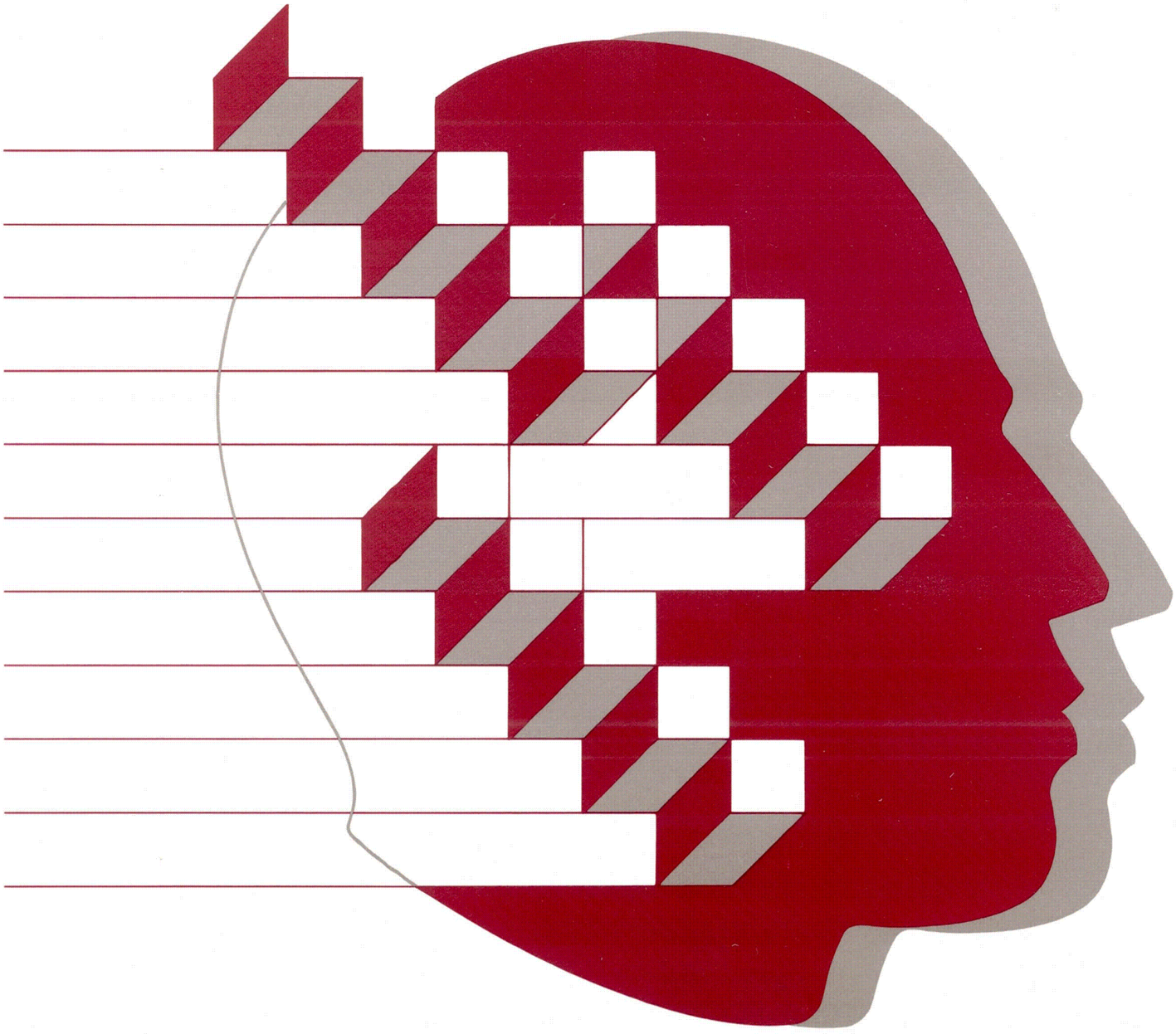
HP 3000 Computer Systems

HEWLETT PACKARD

HPToolset Reference Manual



HP 3000 Computer Systems

HPToolset

Reference Manual

HEWLETT PACKARD

370 WEST TRIMBLE ROAD, SAN JOSE, CALIFORNIA 95131

Part No. 32350-90001
U0184

Printed in U.S.A. 7/82
01/84

PRINTING HISTORY

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by the customer. The dates on the title page change only when a new edition or a new update is published. No information is incorporated into a reprinting unless it appears as a prior update; the edition does not change when an update is incorporated.

The software code printed alongside the date indicates the version level of the software product at the time the manual or update was issued. Many product updates and fixes do not require manual changes and, conversely, manual corrections may be done without accompanying product changes. Therefore, do not expect a one to one correspondence between product updates and manual updates.

First Edition	JUL 1982	32350A.00
Second Edition	JAN 1984	32350A.01

LIST OF EFFECTIVE PAGES

The List of Effective Pages gives the date of the most recent version of each page in the manual. To verify that your manual contains the most current information, check the dates printed at the bottom of each page with those listed below. The date on the bottom of each page reflects the edition or subsequent update in which that page was printed.

Effective Pages	Date
All	JAN 1984

PREFACE

Throughout this reference manual, the term Toolset is used to refer to the HPToolset software.

Toolset is a software package designed to make it easier for Cobol and Pascal programmers to develop application programs.

There are three pieces of Toolset documentation:

(1) the self-study course was developed for the novice who requires step by step instruction in how to use the product. It is intended to be read consecutively.

(2) the Reference Manual was developed for the experienced programmer who requires a technical reference to the product. As such, it is designed to be used for discrete consultation and not for consecutive reading. And, lastly, there is

(3) the SE Training Course for Systems Engineers.

The Reference Manual contains eight sections on the following topics:

Section 1	An Introduction to the key features of the product
Section 2	The Function Key Sets
Section 3	A brief alphabetical summary of all Toolset commands and function keys.
Section 4	A detailed alphabetical directory of all Toolset commands and function keys.
Section 5	The Workspace concept and how files are managed
Section 6	Editing
Section 7	Program Translation: Prep and Compile
Section 8	Symbolic Debug

CONVENTIONS USED IN THIS MANUAL

NOTATION	DESCRIPTION
`uppercase`	Words in syntax statements which are in uppercase must be entered exactly as shown. Punctuation characters other than brackets, braces and ellipses must also be entered exactly as shown. For example: `EXIT;`
`lowercase`	Words in syntax statements which are in lowercase denote a parameter which must be replaced by a user-supplied variable. For example: `CLOSE` *filename*
[ ]	An element inside brackets in a syntax statement is optional. Several elements stacked inside brackets means the user may select any one or none of these elements. For example:  User *may* select A or B or neither.
| |	Optional keywords will be enclosed by verticle bars.
{ }	When several elements are stacked within braces in a syntax statement, the user must select one of those elements. For example:  User *must* select A or B or C.
. . .	A horizontal ellipsis in a syntax statement indicates that a previous element may be repeated. For example: [,*itemname*]. . . ; In addition, vertical and horizontal ellipses may be used in examples to indicate that portions of the example have been omitted.
,	A shaded delimeter preceding a parameter in a syntax statement indicates that the delimeter *must* be supplied whenever (a) that parameter is included or (b) that parameter is omitted and any *other* parameter which follows is included. For example: *itema*[,*itemb*][,*itemc*] means that the following are allowed: *itema* *itema*,*itemb* *itema*,*itemb*,*itemc* *itema*,,*itemc*

CONVENTIONS (continued)

underlining	When several choices are given for a parameter, the default may be underlined.
shading	Shading represents inverse video on the terminal's screen. In addition, it is used to emphasize key portions of an example.

CONTENTS

Section 5
WORKSPACE & FILE MANAGEMENT

Section 6
EDITING

Section 7
PROGRAM TRANSLATION

Section 8
SYMBOLIC DEBUG

INTRODUCTION

SECTION 1

TOOLSET: ABOUT THE PRODUCT

Purpose

Toolset is designed to facilitate three core program development activities: 1) coding, 2) compiling and 3) debugging.

Design Features

To accomplish these objectives, it has six key design features.

The first three are general management and guidance features:

- The User Interface enables you to access and communicate with Toolset through the use of commands and function keys.
- The on-line Help Facility is an introduction to Toolset. It can be used as such or as a quick reference. The Help Facility contains seven primary categories of information:

  Softkeys (function keys)
  Commands
  Exec
  Workspace
  Edit
  Program Translation
  Debug

- The Workspace controls all of the files that are used in the development of a single COBOL II or Pascal program:

  Source files
  Listing files
  USL file
  Program file

The coding specific features of Toolset are the Editor and the Program Translator:

- The Editor allows you to easily compose and modify your programs on the terminal screen.
- The Program Translator does the Compilation and Preparation activities for your program.

As the name indicates, Symbolic Debug is the debugging feature of Toolset.

- It allows you to interactively debug your program without knowing memory locations or code addresses.

HOW TO RUN TOOLSET

To access Toolset, type the MPE command -

```
:Run Toolset.Pub.Sys
```

To exit Toolset Press the Exit function key or type the EXIT command after the Toolset prompt:

```
>>EXIT
```

THE WORKSPACE CONCEPT

A Workspace is an environment in which you develop a single program. It is also a file directory which contains all of the source, listing, USL and program files used for your program. The Workspace has the file code TSR when displayed with the :LISTF command.

Creating a Workspace

Almost all Toolset operations require a defined workspace, so upon logging on and accessing Toolset, you must create a workspace or declare a workspace that is already defined. To do this, either press the Workspace function key or enter the WORKSPACE command following the >> prompt on your screen.

If the Workspace name which you enter does not already exist within the system, Toolset will create the new Workspace as a result of the following dialogue:

```
>>WORKSPACE ONE

--->Create Workspace One?  YES

Default language for one?  COBOL
```

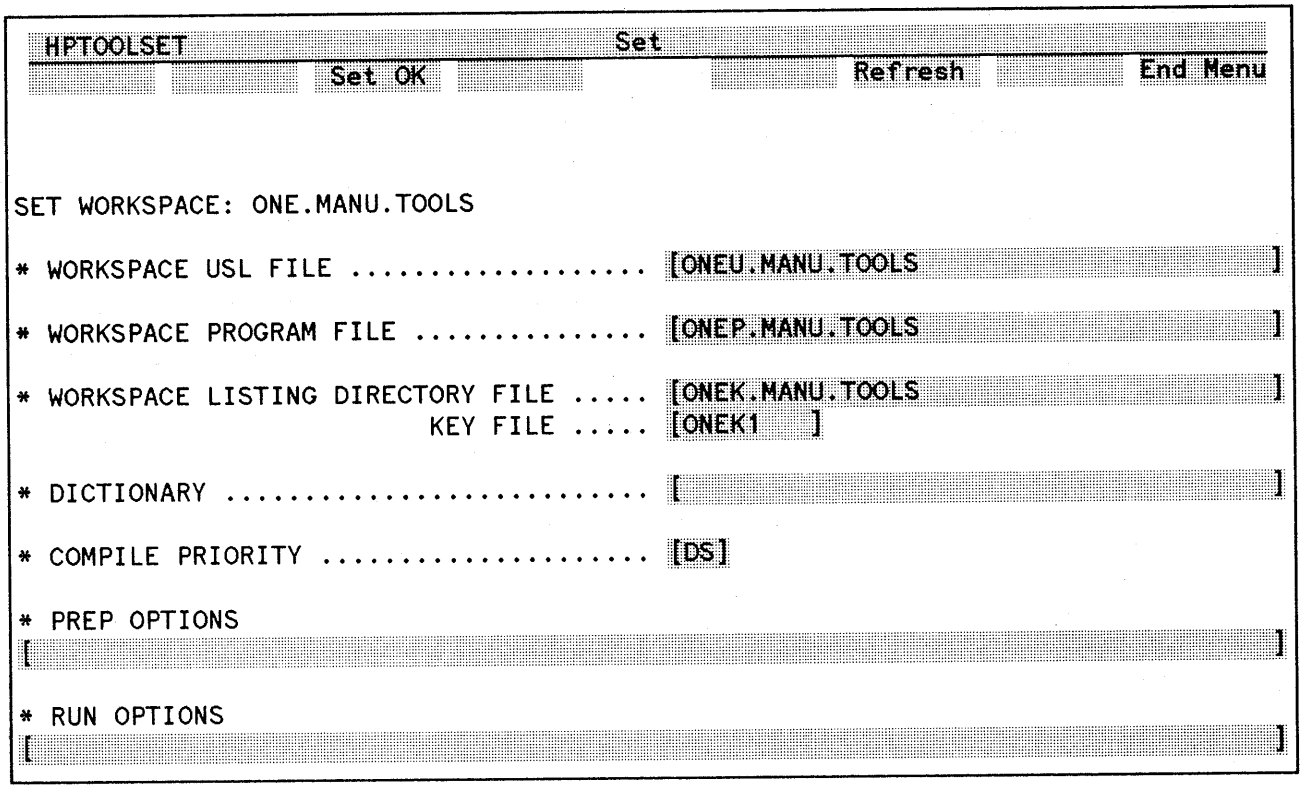


Figure 1-1. Defining a Workspace

When you create a Workspace, the Set Workspace menu of the default options is displayed. Toolset sets up default USL and program filenames by appending a U or a P to the first seven characters of the Workspace name which designates the files for program translation and execution. The default compile priority queue is DS. Any of the defaults may be changed. If you choose to supply your own, the filenames must be unique. When you have selected all of your options, press the Set OK function key. The default SET options for Workspace One are:

```
USL --> OneU.group.account

Prog --> OneP.group.account

Listing Directory  OneK.group.account

        Key file   OneK1

Dictionary --> left blank.   DICT.PUB will be used by default.

Priority --> DS

Prep options --> This field is left blank by default; OneU is prepped
                 into OneP with MPE default options.

Run options --> This field is left blank by default; OneP will be run
                with MPE default options.
```

TOOLSET MODES OF OPERATION

You can communicate with Toolset by pressing a Toolset function key or entering a command. When you do so, the terminal is in one of three modes: Command, Visual or Menu.

Command Mode

Purpose	Command mode is the data entry mode which immediately follows the logon and is indicated by the >> prompt.
How to Enter	You are in Command mode when you enter Toolset. You can reenter it from Visual Mode by pressing the Command Mode, End Edit, End Read or End List function key. From Menu mode, you can reenter command mode by pressing the End Menu or Set Ok key.

Visual Mode

Purpose	Allows you to edit or read files
How to Enter	You are in Visual mode when you are in the Toolset Editor or reading a file. To access the Visual mode editor from Command mode or Menu mode, press the Edit function key or type EDIT. To read a file in Visual mode, press the Read or Listing key or type the corresponding command.

Menu Mode

Purpose	The menus Display a variety of information such as defaults and previously defined options. They also allow you to define and change options.
How to Enter	Menu Mode is accessed through the SHOW, SET and HELP commands.

The Help Facility

To get the Help Facility press the Help key or type the HELP command. You will then see the Toolset Help Overview screen. It describes Toolset and lists seven options for further description of its basic features:

Softkeys
Commands
Exec
Workspace
Editor
Program
Debug

There are three basic levels of explanation within the Help Facility: Overview, Toolset Features and Command Descriptions. At each level you will be given a list of options. When you select an option, Toolset takes you to the next level of explanation. If you want to return to the previous level of explanation, simply press the Overview key.

Level 1: TOOLSET OVERVIEW

Level 2: FUNCTION KEYS	COMMANDS	EXEC	WORKSPACE
Level 3:			
Cmd Mode	See	END	Convert
Clrmark	Section 2	EXIT	Copyfile
Debug	Table 2.1	HELP	Discard
Lib Keys	Summary of	REDO	Label
Main Keys	All Toolset	SHOW	List Change
Mark	Commands	XEQ	Purge
Nextkeys			Recover
Nextpage			Rename
Prevkeys			Restore
Prevpage			Setref
Refresh			Setversion
			Store
			Use
			Workspace

Level 2:	EDITOR	PROGRAM	DEBUG
Level 3:	ADD	COMPILE	AT
	CHANGE	FIND	BREAK
	COPY	ERROR	CALLS
	DELETE	GO	CLEAR
	EDIT	LISTING	DATATRACE
	FIND	PREP	DISPLAY
	GENERATE	PRINT	MOVE
	LIBCOPY	RUN	RESUME
	LIBEDIT	SHOW SOURCE	RETRACE
	LIBLIST		SYSDEBUG
	LIBPURGE		TRACE
	LIST		
	MODIFY		
	MOVE		
	READ		
	RENUMBER		
	SHIFT		
	UNDO		

MPE Commands

You can execute MPE commands without exiting Toolset by typing a colon (:) before the command. Most commands which are preceded by a colon are passed to MPE for execution. The MPE commands which can be entered from Toolset are:

ALTSEC	SECURE
BUILD	SEGMENTER
COMMENT	SETDUMP
FILE	SETJCW
GETRIN	SETMSG
HELP	SHOWDEV
LISTF	SHOWIN
LISTVS	SHOWJCW
PREP	SHOWJOB
PTAPE	SHOWME
PURGE	SHOWOUT
RELEASE	SHOWTIME
RENAME	SPEED
REPORT	STREAM
RESET	TELL
RUN	TELLOP

Consult the MPE Command Reference Manual for the correct syntax of the above commands.

HARDWARE REQUIREMENTS

Processor

HPToolset runs on the HP-3000 series III computer systems or larger that can support MPE IV with a minimum of one Meg of main memory. Specific hardware requirements for each system depend on the application environment and expectations for reaponse time.

Memory Requirements

HPToolset requires a minimum of one Meg of main memory and approximately five thousand sectors of disc memory for the three program files, message files, error message catalog, HELP catalog, softkey labels and fasts form files.

Terminal Requirements

To provide the needed visual and interactive user interface, a VPLUS compatible terminal is required to fully support all the screen functions of HPToolset.

FUNCTION KEY SETS

SECTION 2

DESIGN FEATURES

Toolset function keys are constructed in a looping manner. The NextKeys and PrevKeys are available to move within the loops. Pressing some of the keys in a particular loop will cause Toolset to branch to other loops.

LOCATION OF FUNCTION KEYS

When using terminals in the 264x family, function key labels appear at the top of your screen. On 262x terminals, the function keys are displayed at the bottom of the screen.

FUNCTION KEY SETS

The following are the main function key sets. The original set allows you to define environment options, declare a workspace and exit Toolset. Once you have declared a workspace, you will be in the main loop.

Workspace

```
Set Env          Workspac                  Refresh   Help   Exit
```

The Main Loop contains three function key sets.

Main Edit (Edit Functions)

```
ShoFiles  Read  PrevKeys NextKeys  Edit  Refresh  Help  Exit
```

Main Program 1 (Program Functions)

```
ShoSourc Listing PrevKeys NextKeys Compile  Prep  Run  Exit
```

----OR----

Main Program 2 (Program Functions)

Is displayed only when you are running a Symbolic Debug program.

```
ShoSourc Listing PrevKeys NextKeys Compile DebugKey          Exit
```

Main Utility (Workspace Functions)

```
Set Env  Show Eq  PrevKeys NextKeys Copyfile Lib Keys Set Work  Exit
```

The Lib Keys Set is accessed through the Libkeys branching key in the Mainloop. It looks like this:

```
Set Lib  Show Lib Libpurge Liblist  Libedit  Refresh  Libcopy  Mainkeys
```

These are the Editor function key sets. They are displayed in Visual mode only.

Edit Find

```
  Find    Change  PrevKeys NextKeys  Cmd Mode Listing    Help    End Edit
```

Edit Browse

```
PrevPage NextPage PrevKeys NextKeys  Find -N  Find +N   Set N   End Edit
```

Edit Move

```
  Mark    Clr Mark PrevKeys NextKeys   Move     Copy     Undo    End Edit
```

Edit Add

```
  Mark    Clr Mark PrevKeys NextKeys   Add    Generate           End Edit
```

Edit Print

```
Set Edit  Print   PrevKeys NextKeys  Cmd Mode Refresh   Undo    End Edit
```

The Visual Read Loop has only two sets. They are as follows:

Read Find

```
  Find    Print              NextKeys  Cmd Mode Refresh   Help    End Read
```

Read Browse

```
PrevPage NextPage            NextKeys  Find -N  Find +N   Set N   End Read
```

The Debug loop also has only two sets. They are:

Debug Edit

ShoDebug	Listing	Edit	NextKeys	Resume	Refresh	Help	MainKeys

Debug Utility

Trace	Retrace	Calls	NextKeys	Resume	At Next	Clear	End Run

The Visual Listing loop has four function key sets. The first set will appear only if a program is being run with symbolic debug when the listing is displayed.

Listing Debug (Debug only)

Mark	Clr Mark	PrevKeys	NextKeys	Resume	At	Display	End List

Listing Find

Find	Print	PrevKeys	NextKeys	Cmd Mode	Refresh	Help	End List

Listing Browse

PrevPage	NextPage	PrevKeys	NextKeys	Find -N	Find +N	Set N	End List

Listing Edit

Mark	Clr Mark	PrevKeys	NextKeys	Edit	Find Err	Listing	End List

Help

PrevPage	NextPage	SelectOK		Overview	Refresh	Help	End Help

Set

		Set OK			Refresh		End Menu

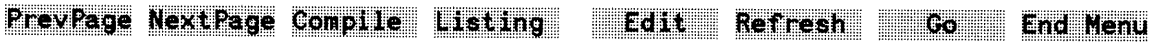
Show

		ShowOK			Refresh		End Menu

Show Equates

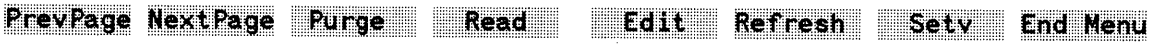
The following set is displayed when you are in the Show Source menu.

Show Source



This set is displayed in the Show Files menu.

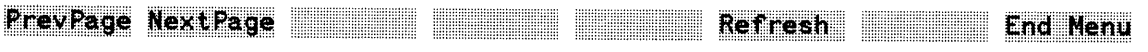
Show Files



Show Debug

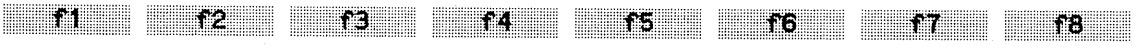
The Show Debug Set is accessed through the SHOW DEBUG command or the ShoDebug function key. This set is accessed through the Cmd List key in the SHOW DEBUG set.

Command List



A blank set is shown when running programs other than the Symbolic Debug Program.

Blank Set



SUMMARY OF COMMANDS AND FUNCTION KEYS

SECTION 3

SUMMARY OF TOOLSET COMMANDS

ADD Enables you to add text to the file you are currently editing.

```
A[DD] [line number] [BY increment]
```

AT Sets a breakpoint in a user program.

```
AT {location} [ |EVERY| n |TIMES| ] [FOR {n   } |TIMES|]
   {   NEXT }                            {ALL}

   [DO   [command-list ] ]
         [(BREAK)      ]
```

BREAK Is used in the command list operand of the AT or DATATRACE command. Returns control to the command interpreter.

```
B[REAK]
```

CALLS Displays the names of the main program and currently called subprograms (COBOL) or procedures (Pascal).

```
CA[LLS]
```

CHANGE Allows you to change specific strings or column ranges in the edit file.

```
CH[ANGE][    [[ALL] find string]  |TO| change string [|IN| line range]    ]
        [    [col pos[:col pos]]                                          ]
```

CLEAR Removes a breakpoint in the user program.

```
        {location}
CL[EAR] {  NEXT  }
        {  ALL   }
```

COMPILE Causes Toolset to call the appropriate compiler for your COBOL or Pascal source file.

```
COM[PILE] [sourcefile[version designator] [uslfile]
          [,sourcefile[version designator] [uslfile]]...]
```

CONVERT Converts files from ASCII or KSAM to TSAM format or from TSAM to ASCII format.

```
CON[VERT] filename1[version designator] [|TO| filename2]
```

COPY Copies text to a file, from a file or from one location to another within the edit file.

```
COP[Y] {char range                  [TO filename     ] [BY increment]}
                                    [TO char position]

   {filename[vers des][char range][TO char position] [BY increment]}
```

COPYFILE Copies a named TSAM file to a new file in the current Workspace.

```
COPYF[ILE] [filename1[version range][|TO| filename2 [N[EWVERSION]]]]
```

DATATRACE Monitors the value of a data item in a user program.

```
DA[TATRACE] data-item[DO command-list[NOMESSAGE] ]
                     [OFF                        ]
```

DELETE Deletes text from an edit file.

```
D[ELETE] character rangelist
```

DISCARD Cancels a USE command.

```
DISC[ARD] filename1[version designator] [,filename2[version designator]]...
```

DISPLAY Causes the contents of a specified data item in a user program to be displayed on the terminal screen.

```
                                    [ O[CTAL]       ]
          {rec-item [USING t1[,t2...]] [ I[NTEGER]     ] [|FOR| n |ITEMS|]}
DI[SPLAY] { data-item                  [ C[HARACTER]   ]
          {"literal"  }                [ H[EXADECIMAL]]
```

EDIT Opens a file for editing.

```
ED[IT] [filename] [linepos]
```

END Terminates a Toolset function or activity.

```
EN[D]  [A[LL]          ]
       [E[DIT]         ]
       [L[ISTING]      ]
       [RE[AD]         ]
       [RU[N]          ]
       [C[OMPILE]      ]
```

EXIT Exits Toolset and returns you to MPE.

```
EX[IT]
```

FIND Locates a line of text or string in the file you are editing or reading.

```
F[IND] [  [ALL] find-string     [ |IN| character rangelist] ]
       [   line position                                     ]
       [level1[#line position]                               ]
```

GENERATE Generates COBOL and Pascal data declarations from information in the data dictionary.

```
          F[ILE]     [NAMELIST] [V[AR]       [line #]          ]
GENERATE  E[LEMENT]  [NAMELIST] [T[YPE]      [line #]          ]]][BY Increment]
          I[MAGE]                [[ENV line #]][DATA [line #]]
          C[OMAREA]              [line #                       ]
```

HELP Accesses the Toolset Help facility.

```
H[ELP]  [OVERVIEW
        [function name
        [
        [command name      [SYNTAX     ]
        [                  [PARAMETERS ]
        [                  [OPERATION  ]
        [                  [EXAMPLE    ]
```

LABEL Allows you to enter a comment for a file version.

```
LA[BEL] filename[version range] "comment string"
```

LIBCOPY Is used to add new modules to a copylib from an existing file.

```
LIBC[OPY] [filename] [line range] [TO modname[OF libname]]
```

LIBEDIT Allows you to edit a module or create a new module in an existing copylib.

```
LIBE[DIT] [modname [OF libname] [line#]]
```

LIBLIST Allows you to display the contents of a copylib without having to convert it from KSAM to TSAM format.

```
LIBL[IST] [modname [OF libname]] [|TO| LP] [UN[NUMBERED]]
```

LIBPURGE Is used to delete one or more modules from a copylib.

```
LIBP[URGE] [modname[,modname2...] [OF libname]]
```

LIST Allows you to list a specific portion of your edit, read or listing file while in command mode.

```
L[IST] line rangelist [|TO| LP] [UN[NUMBERED]]
```

LIST CHANGE Provides a list of the changes between an active version of a source file and the next version of the same source file.

```
L[IST] C[HANGE] filename[version range] [|TO| LP]
```

LISTING Displays a compile listing.

```
          Program-ID
LISTI[NG] [          ] [Stmt#]
          Procedure
```

MODIFY Allows you to change records in your edit file while you are in command mode.

```
M[ODIFY] line rangelist
```

MOVE (Edit) Moves text from one location to another.

```
MOV[E] character range |TO| character position [BY increment]
```

MOVE (Debug) Transfers the value of a literal, figurative constant or data item to another data item.

```
       {literal     }
MOV[E] {data-item-1 }  TO data-item-2 [|FOR| n |ITEMS| ]
       {fig-constant}
```

PREP Prepares a USL file and creates a program file.

```
PREP [uslfile, progfile]

      [;ZERODB]
      [;PMAP]
      [;MAXDATA=segsize]
      [;STACK=stacksize]
      [;DL=dlsize]
      [;CAP=caplist]
      [;RL=filename]
      [;PATCH=patchsize]
      [;NOSYM]
      [;FPMAP] [;NOFPMAP]
```

PRINT Sets the flag to determine whether or not hard copy listings are created during a compile.

```
PRI[NT] {ON }
        {OFF}
```

PURGE Purges files that are owned by the current workspace.

```
PU[RGE]  {filename1[version range]      [,filename2[version range]]...  }
         {@                                                            }
         {WORKSPACE[workspacename]                                     }
```

READ Allows a Toolset formatted file to be read without being open for editing.

```
REA[D] [filename[version designator]] [lineposition]
```

RECOVER Recovers all versions of a file.

```
REC[OVER] [filename[,U[NLOCK]]]
```

REDO Allows you to correct and re-execute the last command or command list.

```
RED[O]
```

REFRESH Redisplays the screen if it was accidentally destroyed.

```
REF[RESH]
```

RENAME Changes a file's identification within the system.

```
RENA[ME] filename1|,|filename2
```

RENUMBER Renumbers all the lines of text in your current edit file.

```
RENU[MBER] [BY increment]
```

RESTORE Restores a workspace and all files owned by it from tape.

```
REST[ORE] workspacename [KEEP]
```

RESUME Restarts the user program execution.

```
RES[UME] [para-name     ]
         [section-name  ]
```

RETRACE Lists the last N COBOL paragraphs or Pascal procedures executed in the user program.

```
                |PARAGRAPHS|
RET[RACE] [n                ]
                |PROCEDURES|
```

RUN Executes a program file.

```
RUN [progfile] [,entrypoint]

     [;NOPRIV]
     [;LMAP]
     [;DEBUG]
     [;MAXDATA=segsize]
     [;PARM=parameternum]
     [;STACK=stacksize]
     [;DL=dlsize]

           G
     [;LIB={P}]
           S

     [;NOCB]
     [;INFO=string]

               [*formaldesig] ]                [*formaldesig ] ]
     [;STDIN = [fileref     ] ] [;STDLIST = [fileref [,NEW] ]
               [$NULL       ] ]             [$NULL        ] ]
```

SET EDIT Allows you to view and change edit options that you defined when you created your edit file.

```
SET ED[IT]
```

SET ENVIRONMENT Allows you to change the design features such as page size.

```
SET EN[VIRONMENT]
```

SET LANGUAGE Changes the workspace default language.

```
SET LAN[GUAGE]      [|=| C[OBOL]      ]
                    [    P[ASCAL]     ]
                    [    O[THER]      ]
```

SET LIBRARY Is used to create a copylib and set the default library name for copylib editing commands.

```
SET LIB[RARY] [|=| libname]
```

SETREF Designates the Reference version of a source file.

SETR[EF]filename1[version designator1][,filename2[version designator2]]...

SETVERSION Freezes changes to a file and increments the latest version number.

SETV[ERSION] [filename]

SET WORKSPACE Presents the Set Workspace menu of options.

SET W[ORKSPACE]

SHIFT Allows you to shift a block of text to a particular column and shift text right or left a specified number of characters.

```
SHI[FT] line rangelist [|FROM| column number] {|TO| column number}
                                               { L number         }
                                               { R number         }
```

SHOW ACTIVITIES Lists all active functions in your Toolset sessions.

SHO[W] A[CTIVITIES]

SHOW DEBUG Displays all active breakpoints and datatrace variables.

SHOW D[EBUG]

SHOW EQUATES Displays the file equations that have been set up by the USE command.

SHOW E[QUATES]

SHOW FILES Displays a list of all owned and used files for a particular Workspace.

SHO[W] F[ILES] [workspacename]

SHOW LABELS Displays any comments associated with a version of a TSAM file.

SHO[W] LA[BEL] filename[version range]

SHOW LIBRARY Displays the file information about a copylib and the names of the modules within it.

SHOW LI[BRARY] [libname]

SHOW SOURCE Displays Workspace source files.

SHO[W] S[OURCE]

STORE Stores the entire Workspace onto tape.

ST[ORE] [workspacename]

SYSDEBUG Enters MPE Debug when executing a user program.

SYS[DEBUG]

TRACE Identifies each subprogram, section, paragraph or procedure just before it executes.

T[RACE] [OFF]

UNDO Allows you to cancel the last CHANGE, DELETE, MOVE, SHIFT or MODIFY command just entered.

UN[DO]

USE Allows two Workspaces to share ownership of a file and specifies which version of the file will be shared.

USE filename1[version designator] [,filename2[version designator]] ...

USE [formal1 |FOR|] filename1[version designator]
[,[formal2 |FOR|] filename2[version designator]] ...

WORKSPACE Allows you to create and change Workspaces.

W[ORKSPACE] [workspacename]

XEQ Allows input from a specified file.

X[EQ] filename[version designator]

SUMMARY OF TOOLSET FUNCTION KEYS

Add	Enables you to add text to the file you are currently editing.
At	Sets a breakpoint in a user program.
Change	Allows you to change specific strings in the edit files.
Clear	Removes the current breakpoint in the user program.
Clear Mark	Clears all active marks for the current process that have been set with the mark key.
Command List	Displays the commands to be performed when a breakpoint occurs or data trace variable is displayed.
Command Mode	Returns you to command mode from visual mode without closing the current edit, read or listing file.
Compile	Causes Toolset to call the appropriate compiler for your COBOL or Pascal source file.
Copy	Allows you to copy text from one position to another within an edit file.
Copyfile	Copies a named TSAM file into the current workspace.
DebugKey	Branches from the main function key loop to the debug.
Display	Causes the contents of the Marked data item in a user program to be displayed on the terminal screen.
Edit	Opens a file for editing.

End Edit	Ends the edit function, closes the file and puts you in command mode.
End List	Ends the listing function, closes the file and puts you in command mode.
End Menu	Returns you to the main set of function keys.
End Read	Ends the read function, closes the file and puts you in the command mode.
End Run	Terminates the execution of your program.
Exit	Exits Toolset and returns you to MPE.
Find	Locates a line of text or string in the file you are editing or reading.
Find Err	Displays the compile time errors and warning messages.
Find -N	Finds the line that is N lines before the first line displayed on the screen.
Find +N	Finds the line that is N lines after the first line displayed on the screen.
Go	Activates a one step method of compiling, prepping and running your source file(s).
Help	Accesses the Toolset Help facility.
Libcopy	Is used to add new modules to a copylib from an existing file.
Libedit	Allows you to edit a module in an existing copylib.
LibKeys	Is a branching function key by which Copylib editing function keys may be reached.

Liblist	Allows you to display the contents of a copylib without having to convert it from KSAM to TSAM format.
Libpurge	Is used to delete one or more modules from a copylib.
Listing	Displays the compile listing for the most recently compiled source file.
MainKeys	Is located in the Debug function key loop and Copylib editing set and returns you to the Main function key loop.
Mark	Allows you to mark one line or character of text.
Move (Edit)	Moves text from one location to another.
NextKeys	Erases your current set of function keys and displays the next set in the particular loop.
NextPage	Erases the existing screen and displays the next one in the current file.
Prep	Prepares the workspace USL file and creates the workspace program file.
PrevKeys	Erases the existing set of function keys and displays the previous set in the particular loop.
PrevPage	Erases the existing screen and redisplays the previous one.
Print	Generates a hard copy of the currently displayed listing or source file.
Purge	Purges files that are owned by the current workspace.
Read	Allows a Toolset formatted file to be read without being open for editing.
Refresh	Redisplays the screen if it was accidentally destroyed.

Resume	Restarts the user program execution.
Retrace	Lists the last N COBOL paragraphs or Pascal procedures executed in the user program.
Run	Executes the workspace default program file.
Set Edit	Allows you to view and change edit options that you defined when you created your edit file.
Set Environment	Allows you to change the design features such as page size.
Set Library	Is used to create a copylib and set the default library name for copylib editing commands.
Set Workspace	Presents the Set Workspace menu and options.
Show Debug	Displays all active breakpoints and datatrace variables.
Show Equates	Displays the file equations that have been set up by the USE command.
Show Files	Displays a list of all owned and used files for the current Workspace.
Show Library	Displays the file information about the default copylib and the names of the modules within it.
Trace	Identifies each subprogram, section, paragraph or procedure just before it executes.
Undo	Allows you to deactivate the modifications which were incorporated the last time Return or a function key was pressed.
Workspace	Allows you to create and change workspaces.

TOOLSET COMMANDS AND FUNCTION KEYS

SECTION 4

ADD

Enables you to add text to the file you are currently editing.

Syntax

```
A[DD] [line number] [BY increment]
```

Parameters

Line Number	Allows you to insert text at a specified place in your file. Defaults to LAST if you do not include a line number.
BY Increment	Allows you to indicate an increment for the lines you are adding.

Discussion

Text can be added at a specific line or, if no line number is specified, at the end of your file (default). If the file you are adding to is empty, you will be prompted with the first line number. A new line number will be displayed after each carriage return.

Before starting the add, the line prior to your addition will be displayed on your screen. You will be prompted with the next line number and may add text at that point. Lines are added according to the particular increment value you have indicated. You can define a default increment in the SET EDIT menu. When adding between existing lines of text, that default may be overridden with a smaller number to allow more lines to be added. The text you add can only be input from $STDIN and not from an XEQ file or a command list. When you enter Add Mode, any pending XEQ files or command lists will be terminated.

To end Add mode, press control-y, enter a double slash (//) or press any function key. When Add mode is ended you are in Command mode.

When you are adding lines, Toolset will not recognize lines added with your screen edit keys. Text added in this way will be entered at the next add line after you press the return key.

ADD FUNCTION KEY

Allows you to add text to a file you are currently editing.

To use the Add function key:

(1) Display the page where you want to add text
(2) Move the cursor to the line after which you want the added text placed
(3) Press the Mark key twice (line mark)
(4) Press the Add function key to add the text

You will find this key in the Edit Add function key set which is available while editing in visual mode.

Sets the breakpoint in a user program.

Syntax

```
AT {location} [ |EVERY| n |TIMES| ] [FOR { n } |TIMES| ]
   {   NEXT }                        [    {ALL}        ]

   [DO  [ command-list ] ]
        [ (BREAK)      ]
```

Parameters

Location	Paragraph name, section name, subprogram name, procedure name, label name or line number at which you want your program to break. The location may be referenced with an offset such as PROC#25. The breakpoint is set immediately preceding the first executable statement of the location you specify.
NEXT	Specifies a breakpoint to be set at the beginning of the next section, paragraph or procedure and every succeeding section, paragraph or procedure. You cannot abbreviate NEXT. In Pascal, a breakpoint is also set at the end of every procedure.
EVERY n TIMES	Specifies that the breakpoint you set will cause an interruption every nth time it occurs. The interruption occurs before the execution of the statement located at the breakpoint. The default value of n is 1.
FOR n TIMES	This clause specifies that the breakpoint be cleared after it has executed N times. N is an integer. The default is to set a permanent breakpoint.
DO	Specifies that one or more Toolset commands be performed when the breakpoint occurs. The default is to BREAK and prompt you for your command.
Command-list	Takes the form (command [; command] . . .) where the command is a Toolset command.

Discussion

The AT command allows you to set up 15 active breakpoints in your user program. When the breakpoint in your program is reached, control is passed to the DO command list. If there is no DO command list, control is passed to you through the Toolset command interpreter.

You can specify that a Toolset command(s) be performed when a breakpoint occurs by using the DO command list parameter. If one of these commands is the BREAK, RESUME or SYSDEBUG command,

the remainder of the command list will be terminated when that command is executed; otherwise, after all the commands in the list have been executed, Symbolic Debug will execute a **BREAK** command.

Alias (Pascal)

A level 1 procedure with an alias name should always be referenced by its alias name. A level n procedure should always be referenced by its originally declared name, even if it has an alias.

Private Proc (Pascal)

Breakpoints can also be set symbolically by specifying any procedure name that is in the PMAP listing. The procedure must be compiled and prepared with Symbolic Debug. If the Private Proc option is set OFF, breakpoints can be set by referencing level n procedure names without level 1 procedure qualification. If a level n procedure has an alias and Private Proc is set OFF, you can set a breakpoint by simply referencing the alias name. If you qualify the level n procedure name, it must be referenced by its originally declared name and not it's alias.

Statement Number/Current Procedure (Pascal)

If you are setting a breakpoint at a location within the current procedure, only the statement number needs to be specified. If the breakpoint you are setting is outside of the current level 1 procedure, the procedure or label name must be used as well.

Last Procedure Statement Number (Pascal)

Breakpoints cannot be set at the last statement number of a procedure. If the last END statement number is the same as the BEGIN statement number of a containing procedure, the breakpoint will be set at the beginning of the containing procedure and not at the end of the contained procedure.

Example

```
Symbolic Debug Command

>>AT #14

Listing File

00000 PROCEDURE L1;
00001 PROCEDURE L2;
00002    BEGIN
          .
          .
          .
00013       A: =5;
00014       END;    <---- Breakpoint is not set here

00014  BEGIN       <---- Breakpoint is set here
         .
         .
         .

20  END;
```

Setting Breakpoints at Withs

Because the Pascal compiler generates the same statement number for a WITH and its BEGIN statement, a breakpoint set at the WITH will be set at the BEGIN statement.

In the following example, the breakpoint is logically set at statement number 37 immediately before "F2:=15;" is executed.

```
Symbolic Debug Command

>> AT #36
-->BREAKPOINT SET

List File

34   PROCEDURE P1;
35    BEGIN
36      WITH VAR1 DO
37      BEGIN
37          F2 := 15;
38          F1 :=  5;
```

Example

COBOL

(1) >>AT MY-SUBP#112

Sets a breakpoint at line 112 in the subprogram MY-SUBP. The breakpoint can be specified by the paragraph name as in example 2 or by line number as in this example.

(2) >>AT PARA-1 OF MY-SUBP

Sets a breakpoint at paragraph PARA-1 of the subprogram MY-SUBP.

(3) >>AT COMP-COST FOR ALL DO (DISPLAY COST-CNTR; RESUME)

Sets a breakpoint at the paragraph heading COMP-COST in the current program. The breakpoint will occur before the first executable statement following COMP-COST. Every time COMP-COST is executed, the current value of COST-CNTR will be displayed and your program will resume execution.

Pascal

(1) >>AT NEXT DO (DISPLAY F1;BREAK)

This example sets a breakpoint at the beginning and end of the next procedure and every succeeding procedure. AT each breakpoint, the value of variable F1 is displayed and program execution is suspended. The same result would occur if the BREAK was not included in the command list.

If you want execution to automatically resume after the value of F1 is displayed, replace BREAK with RESUME in the command list above.

(2) >>AT PROC1.PROC2.PROC3.PROC4.

Sets a breakpoint at the beginning of the level 4 procedure PROC4.

(3) >>AT PROC1.PROC4

Sets a breakpoint at the beginning of the level4 procedure. One only needs full qualification as shown in example 2 when there is a duplicate procedure name.

(4) Given the following source:

```
$PRIVATE_PROC OFF$
PROCEDURE L1 $ALIAS 'L1ALIAS'$
          PROCEDURE L2 $ALIAS 'L2ALIAS'$
```

and execution is suspended outside of the Level 1 procedure L1. The following AT commands are valid:

```
>>AT L1ALIAS
>>AT L1ALIAS.L2
>>AT L2ALIAS
```

(3) If execution suspended inside procedure L1, the following AT commands are valid:

```
>>AT L1ALIAS
>>AT L1ALIAS.L2
>>AT L2ALIAS
>>AT L1
>>AT L2
>>AT L1.L2
```

AT FUNCTION KEY

Breakpoints can be set by using the compiler listing of your program and the At function key. You can access the At function key when your program has suspended execution and control has returned to Toolset.

Listing Debug Set

To set a breakpoint with the At function key:

(1) Display the compiler listing by pressing the Listing key or typing the command at program execution time.

(2) Position the cursor at the line in your listing where you want a breakpoint to occur and mark it by pressing the Mark key.

(3) Press the At key. A message confirming the breakpoint setting will be displayed at the top of your screen.

Do steps 2 and 3 for each breakpoint you want to set. The program will break immediately before the marked statement is executed and control returns to you. Pressing the At key is the same as using the default condition of the AT command, a permanent breakpoint is set and control is returned to you.

When you have completed setting your breakpoints, press the Resume key and program execution will continue until a breakpoint is encountered or the program terminates.

Debug Utility Set

BREAK

Returns you to Symbolic Debug and the Toolset Command Interpreter from a command list or XEQ file.

Syntax

```
B[REAK]
```

Discussion

The BREAK command is used to return you to Symbolic Debug and the Toolset Command Interpreter from a command list or XEQ file. If Symbolic Debug already has control, entering the BREAK command has no effect. If BREAK is used in a command list or an XEQ file, it terminates the remainder of the list and redirects the Toolset Command Interpreter to accept input from the terminal.

BREAK is used in the command list operand of the AT and DATATRACE commands.

CALLS

Displays the names of the currently executing main and subprograms (COBOL) or names of the main program and currently "called" procedures (Pascal).

Syntax

```
CA[LLS]
```

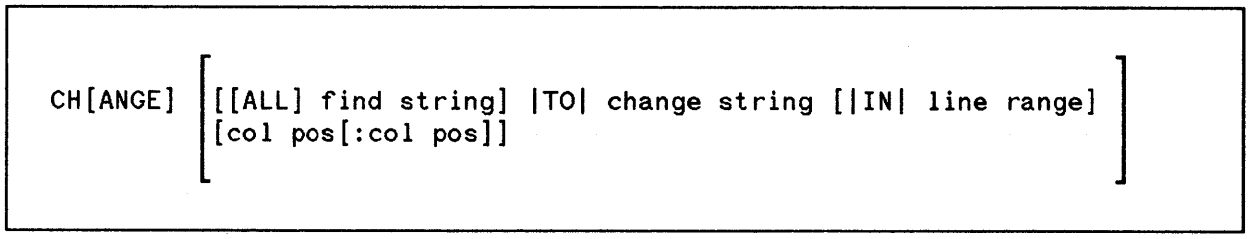
Discussion

The CALLS command displays the names of the main program and the subprograms in COBOL or the currently "called" procedures that are executing if you are using PASCAL. It begins with the most recently "called" procedure. The next statement number to be executed after each call is also displayed.

CHANGE

Allows you to change specific strings or column ranges in the edit file. Can be deactivated with the UNDO command.

Syntax



Parameters

ALL	Causes all occurrences of the find string within the line range to be changed.
Find String	Defines the string of text you want to change. By default, only the first occurrence of the find string will be changed. Strings must be delimited with single or double quotes or brackets. If the string is delimited with double quotes, only occurrences surrounded by one or more spaces or by punctuation will be found. If delimited by single quotes, all occurrences will be found. Strings delimited by brackets affect both upper and lower case occurrences.
Column Position: Column Position	Defines a column range that you want to change. FIRST (F) denotes the leftmost accessible column. LAST (L) denotes the rightmost accessible column. For COBOL files, FIRST represents column 7. For Pascal and other files, FIRST represents column 1.
Change String	Is the replacement text. This string must be delimited by single or double quotes or brackets.
Line Range	The range of lines that will be searched for the find string. By default, the current line will be searched. Has the format n[/n] where n could be a line number, FIRST or LAST.

Discussion

When the CHANGE command works in conjunction with the FIND command, only the CHANGE |TO| string portion of the syntax needs to be used. This method of changing text assumes that a FIND command has been entered. Toolset rememebers the find-string and where it was until a new FIND command is entered or until you specify a new find string in the CHANGE command. The Change function key utilizes this method of defining the find string with the Find key so that when Change is pressed, you need only enter the replacement string. To use the CHANGE command without an associated FIND command, you must enter both the find string and the change string parameters (the string to be changed and the replacement string).

When the Change command is entered without any parameters, TOOLSET prompts for the change string. If there is a previous change string, it is displayed in parentheses and can be reused by pressing the carriage return.

If the change causes a line of text to be longer than the record length of your file, the line will be truncated.

Example

```
>>CHANGE ALL 'ARRAY'  'RECORD' IN 50/75
   54    ABC      :    RECORD [1. .10] OF CHAR:
   58    INTRECORD :      ARRAY [1..100] OF INTEGER;
   58    INTRECORD :      RECORD [..100] OF INTEGER;
   63    ERR_CAUSE :  RECORD[1..5] OF PACKED ARRAY[1..60] OF CHAR;
   63    ERR_CAUSE :  RECORD[1..5] OF PACKED RECORD[1..60] OF CHAR;
>>

>>CHANGE 15:20 TO "MOVES" IN 200/210
  200              MOVES FILE1 TO FILE1X
  201              MOVES FILE2 TO FILE2X
  202              MOVES FILE3 TO FILE3X
  203              MOVES FILE4 TO FILE4X
  204              MOVES FILE5 TO FILE5X
  207              MOVES FILE10 TO FILE10X
  208              MOVES OLDFILE TO NEWFILE
>>

>>FIND 'OLD' IN ALL
  208              MOVES OLDFILE TO NEWFILE
>>CHANGE TO 'NEW'
  208              MOVES NEWFILE TO NEWFILE
>>
```

CHANGE FUNCTION KEY

Allows you to change specific strings in the edit file while in visual mode. When you press the Change key, the last change-string that you entered will be displayed in the message window on your screen. This assumes that you have already found a find-string with either the FIND command or function key. The previous change will be repeated if you press return. If you want to redefine the change string, type a new string and press return.

CLEAR

Removes a breakpoint in the user program.

Syntax

```
CL[EAR] {location}
        {NEXT    }
        {ALL     }
```

Parameters

Location	Specifies the location of a set breakpoint that you want to clear.
NEXT	Clears the breakpoint(s) set with a previous AT NEXT command.
ALL	Clears all current breakpoints.

Discussion

The CLEAR command is used to clear some or all breakpoints in a user program. If you enter the command without any parameters, the breakpoint set at the current location in your program will be cleared.

Example

```
>>CLEAR ADD-RT
-->Breakpoint cleared.
```

Clears the breakpoint at location ADD-RT. Toolset displays a confirmation when the breakpoint has been cleared.

CLEAR FUNCTION KEY

Removes a breakpoint in the user program.

Debug Utility Function Key Set

The Clear function key in the Debug Utility Set clears the breakpoint at the current location in your program. If you wish to clear breakpoints from the Show Debug menu, you must first mark the breakpoint by typing any character in the box beside that breakpoint and by pressing the Clear key.

Show Debug Function Key Set

CLEAR MARK FUNCTION KEY

Is available while you are in visual mode. It clears all active marks that have been set with the Mark key for the current process.

This key is valid only while you are doing a visual mode edit or read. To clear marks in the Show Files, Show Debug or Show Source menus type a blank in the box on every line where you wish to clear mark and press enter.

CMD LIST FUNCTION KEY

When you set a breakpoint, or datatrace variable, you may define an associated command list. This causes one or more commands to be performed when that breakpoint occurs or the datatrace variable is displayed. To view the command list associated with the particular breakpoint or datatrace variable, mark the breakpoint while in the SHOW DEBUG menu by entering a character in the box next to it. Then you must press the Cmdlist key. If a breakpoint or datatrace variable has an associated command list, there will be an asterisk in the DO column of the SHOW DEBUG menu.

If you mark more than one breakpoint or datatrace item, only the command list for the first one will be displayed. If you want to view subsequent command lists, press the NextPage function key.

COMMAND MODE FUNCTION KEY

Is available when you are in visual mode. It is used to return to command mode from visual mode without closing the current edit, read or listing file. To return to visual mode, use the EDIT, READ or LISTING commands or function keys.

Causes Toolset to call the appropriate compiler for your COBOL or Pascal source file.

Syntax

```
COM[PILE] [sourcefile[version designator] [uslfile]
      [, sourcefile[version designator] [uslfile]]. . .]
```

Parameters

Source File	The text file from which source records are initially read. The source file(s) must be in TSAM format. If more than one source file is listed, each will compile separately in the order in which they occur.
Version Designator	Specifies the version of your file you want to compile. It takes the form: #n - where n is an integer, Latest (L) or Reference (R) If no version is specified and you own the file, the Latest version is used. The Reference version is used if you share the file.
Uslfile	Allows you to designate the USL file into which the source file will be compiled. The default for both owned and used files is your current Workspace USL file. If your source file is neither owned or used, the default USL is $NEWPASS.

Discussion

When you enter the COMPILE command, it causes Toolset to activate the compiler. The compiler compiles your source file into a USL file. If you enter the COMPILE command without any parameters, you will be prompted for the name of the file you want to compile.

MPE compile commands such as :COBOL and :PASCAL can not be entered from Toolset. You can execute a compile using the :RUN command and file equations. For example:

:FILE COBTEXT = sourcefile

:RUN COBOL II.PUB.SYS;PARM=1

:FILE PASTEXT=sourcefile

:RUN PASCAL.PUB.SYS.;PARM=1

COBTEXT is the COBOL formal file designator and PASTEXT is the Pascal formal file designator. If you compile in this way, errors and listings will not be saved. This method of compilation is the same as exiting Toolset and activating the compiler directly.

COMPILE

You cannot use the :COBOLPREP, :COBOLGO, :PASCALPREP and PASCALGO commands in Toolset.

Example

```
>>COMPILE

-->Name of the file to be compiled?  ZPASPROJ

-->Compiling source file ZPASPROJ.CAROLYN.TOOLS#L
```

COMPILE FUNCTION KEY

Is available in the SHOW SOURCE menu and in the Main Program function key set. To use the Compile key in the SHOW SOURCE menu, designate a source file(s) in the display by typing a character in the box(es) next to the file(s) you want to compile. Then press the Compile key. If you have designated more than one file, each one will compile separately in the order in which they appear. Toolset does not allow simultaneous compiles.

```
 HPTOOLSET                          Show Source
PrevPage NextPage Compile  Listing          Edit   Refresh     Go     End Menu

    SOURCE FILES for Workspace ONE.MANU.TOOLS

     PROGRAM ID/PROCEDURE            LANG     TYPE     SOURCE FILE       LISTFILE

[*]                                  COBOL    MAIN     COMPMAIN
[*]                                  COBOL    SUBP     COMPSUB
[ ]                                  COBOL    INCLUDE  COMPINCL
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
```

Figure 4-1. Designating Files to be Compiled

```
 HPTOOLSET                          Main Program
ShoSourc Listing  PrevKeys NextKeys          Compile    Prep     Run      Exit


     PROGRAM ID/PROCEDURE              LANG     TYPE     SOURCE FILE        LISTFILE

[*]                                    COBOL    MAIN     COMPMAIN
[*]                                    COBOL    SUBP     COMPSUB
[ ]                                    COBOL    INCLUDE  COMPINCL
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
-->Compiling source file COMPMAIN.MANU.TOOLS#L
>>
```

Figure 4-2. Beginning Compile

COMPILE

When you press the Compile key, Toolset will display a message telling you that your file is compiling.

A double arrow prompt will be displayed while the compile is being processed. This means that other Toolset operations can be performed while your source file is being compiled. However, these operations must not access the USL or List file. You can edit your source file at any time other than when you are compiling the LATEST version. During the compile, Toolset does an implicit USE on your source file. The Use appears on the SHOW FILES and SHOW EQUATES menus during the compile. After the compile, an implicit DISCARD is done. When a compile finishes, the compiler statistics, including the number of errors and warnings, are listed on your screen.

From the Listing, you can enter the editor to view the source file and correct errors.

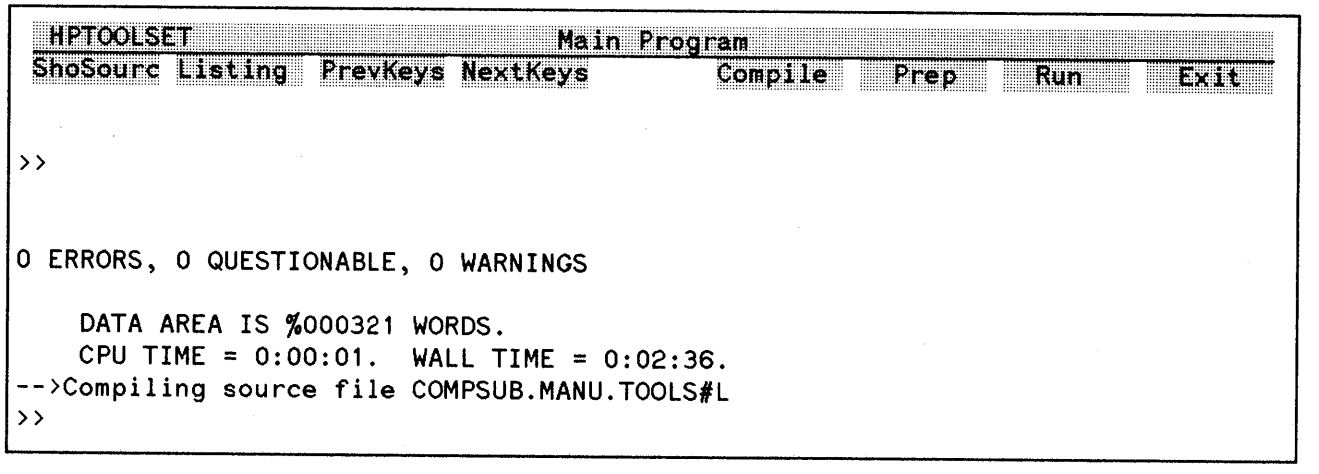


Figure 4-3. Completed Compile

Converts files from ASCII or KSAM to TSAM format or from TSAM to ASCII format.

Syntax

```
CON[VERT] filename1[version designator] [|TO| filename2]
```

Parameters

Filename1	Is the name of the ASCII, KSAM OR TSAM file to be converted.
Version Designator	Applies only to TSAM formatted files. Allows you to specify the version of the TSAM file you are converting. If no version is specified and the from file is owned, the LATEST version will be assumed. If the from file is not owned or is used, the REFERENCE version will be assumed. The Version Designator takes the form #n. N is an integer, #LATEST (#L), or #REFERENCE (#R).
Filename2	The destination file. If this filename is omitted, filename1 will be converted into a temporary file. When the convert is complete, filename1 will be purged and the temporary file will be renamed to filename1.

Discussion

MPE File Conversion

If the MPE file you are converting has a COBOL format filecode or sequence numbers in the first six characters, the TSAM file will be created as a COBOL file with a record size of 74 bytes. Otherwise, Toolset will check the last eight bytes of the record. If they contain only numbers, they will be used as the edit sequence numbers. If the last 8 bytes contain characters other than numbers, Toolset will consider the file unnumbered and assign sequence numbers.

Before the new TSAM file is built, the SET EDIT menu will be displayed to enable you to override any of the default attributes.

Record Size

The default record size for non-COBOL files will be equal to the record size of the old file if it was not numbered. If the old file was numbered, the default record size will be equal to the old record size minus 8. COBOL files will always default to 74 byte records.

File Size

The default file size is one and one half times the number of records in the old file, rounded up to the nearest 100. You should not decrease the file size by more than one third. Otherwise, the file will not be large enough to hold all of the converted records.

TSAM File Conversion

When converting a TSAM file to an ASCII file, you can specify a file equation for the ASCII file to define your own record size, blocking factor, file code, etc. The file equation must specify the fully qualified ASCII file name. If no file equation is specified, the new ASCII file is built with a record size equal to that of the TSAM file plus the sequence number length. The file size is then equal to the active records in the TSAM file, and the blocking factor is calculated to maximize the use of the storage space.

Example

```
(1)  >>CONVERT SSFILE#4 TO TTFILE

Converts version 4 of the TSAM file SSFILE to the
ASCII file TTFILE. SSFILE will be left unchanged.

(2)  >>CONVERT THISFILE

Converts the file THISFILE from one format to another.
When finished, the file will exist only in the new format.
```

COPY

Copies text to a file, from a file and from one position to another within the edit file.

Syntax

```
COP[Y] {char range                    [TO filename      ] [BY increment]}
                                      [TO char position ]

   {filename[vers des] [char range] [TO char position ] [BY increment]}
```

Parameters

Char Range	Defines the range of characters to be copied. If the copy is from another file, the default character range is ALL. Character range is separated by a slash. line position [(column#)] [/line position[(column#)]] or ALL
Filename	The name of the file which is to be copied. This parameter is to be used when copying between files.
Vers Des (Version Designator)	Allows you to copy a particular version of a file. If you do not specify a default, it will be the LATEST version of an owned file, the REFERENCE version of a file which is not owned or a file upon which a USE has been issued.
TO	Defines the destination filename and/or character position. If you do not define a destination file, text will be copied into the edit file.
Char Position	Line position [(column#)]
BY increment	Allows you to copy the text by a specific increment.

Discussion

At the end of a copy, text exists at both the source and the destination. If the copy is being made from one file into the edit file, the file must be a TSAM, numbered ASCII or numbered KSAM file with a record length less than or equal to 132 bytes. If the file that is being copied contains records that are longer than those in the edit file, the lines will be truncated as they are copied.

There must be sufficient space (unused line numbers) to allow the text to be copied. If enough space does not exist, Toolset will not begin the copy.

If you do not specify an increment, Toolset will calculate one.

Example

```
(1)  >>COPY FILE1 TO 103.1
     103.1                    line 1
     103.2                    line 2
     103.3                    line 3
     103.4                    line 4
     103.5                    line 5
     103.6                    line 6
     103.7                    line 7
     103.8                    line 8
     103.9                    line 9
     104                      line 10
     104.1                    line 11
     104.2                    line 12
     104.3                    line 13
     104.4                    line 14
     104.5                    line 15
>>


(2)  >>COPY 355/400 TO 455 BY .01
     455.01                   type
     455.02
     455.03                   lab_engineer=packed array[1..8] of ' '..'Z';
     455.04
     455.05                   sr  =  integer;
     455.06
     455.07                   fix_number  =  integer;
     455.08
     455.09                   stars  =  packed array[1..2] of    .. Z ;
     455.1
 >>

>>COPY 5(14)/5(30) TO 60
   60                  45678901234567890
```

COPY FUNCTION KEY

Allows you to copy text from one position to another within an edit file. The Copy function key is available while in the Visual Editor.

To use it:

(1) Display the text you want copied.

(2) Use the Mark function key to mark the first and last characters or lines you want copied. Press the Mark Key once for character marks and twice for line marks.

(3) Mark the line after which the text should be placed or the character position at which it should be placed.

(4) Press the Copy key.

COPYFILE

Copies a named TSAM file into a new TSAM file in the current Workspace.

Syntax

```
COPYF[ILE] [filename1[version range][|TO|filename2[N[EWVERSION]]]]
```

Parameters

Filename1	Name of the file to be copied.
Version Range	Indicates the range of versions of filename1 which you want to copy. By default, all versions of filename1 will be copied. Unless versions have been purged from the old file, it is faster to COPYFILE all versions than it is to copy a range of versions. Takes the form: #n[/#n] where n is an integer or the LATEST or REFERENCE.
Filename2	Name of the file into which you want to copy. It must be a new file. If a file already exists with this name, you will be asked if it is ok to purge it.
NEWVERSION	Is an optional parameter that renumbers the active versions in filename2 sequentially, starting with version 1.

Discussion

You can copy specific versions of a file or the entire file. If you copy the entire file, identical versions will exist at both the source and destination locations.

The COPYFILE command is a way of establishing ownership of a file that belongs to another workspace. It puts a copy of the file in your workspace, thus allowing you to own that file. Any file that is copied must be in TSAM format.

Example

```
>>COPYFILE OLDFILES#5 TO NEWFILE NEWV

Copies the contents of version 5 of OLDFILE to a new
file named NEWFILE.  The new file has one active version.  It is
version #1 because the NEWVERSION parameter is specified.
```

COPYFILE FUNCTION KEY

Allows you to copy a named TSAM file into the current workspace.

DATATRACE

Monitors the value of a data item.

Syntax

```
DA[TATRACE] data-item [DO command-list [NOMESSAGE] ]
                      [OFF]
```

Parameters

Data-item (COBOL)	Any COBOL data name or index name defined in the COBOL source program. A data item may be qualified with: (1) OF (2) IN (3) subscript (must be an integer value) (4) periods separating the data names where the most significant data name is specified first. You cannot reference a data item in a program other than the one that is currently executing.
Data-item (Pascal)	Any Pascal data name defined in the source program. Standard Pascal selectors are used to qualify arrays and records. In addition, Toolset allows commas when selecting Array elements. OF or IN may be used to select record elements.
DO Command List	Specifies a list of Toolset commands you want automatically executed when the value of a data item changes. Takes the form: (command [; command]...)
NOMESSAGE	Suppresses an identifying message when DATATRACE is executed.
OFF	Disables the DATATRACE. You may also use the Clear function key which is available when the Show Debug menu is displayed.

Discussion

The DATATRACE command saves the value of the data item you specify. Toolset compares the saved value with the current value of the data item at the beginning of each paragraph, section or procedure in your program. If the value has changed, the new value will be saved and a message containing the location and the new value will be displayed. Any commands that you have specified will be executed. If there is no command list, the program will continue running.

The maximum number of data items that can be traced is 3.

If all the commands in your command list have executed, Symbolic Debug will execute a RESUME command. If you include a BREAK or SYSDEBUG command in the command list, the list will terminate.

Pascal

As with COBOL, only 3 data items can be traced at one time in a Pascal program. In addition, a variable cannot be referenced outside of the level of the procedure that it is declared in, nor can it be explicitly turned off when your are working outside of its procedure level.

If you need to set another datatrace and are outside of the procedure where the first three are set, you can:

(1) Go back to the procedure level where the datatraces are set and explicitly turn any or all of the datatraces off.

(2) Enter a SHOW DEBUG command. This produces a menu from which you can mark any or all of the datatraces set and clear them by pressing the Clear function key available when the Show Debug menu is on the terminal screen.

The value of arrays, rows, elements or entire arrays can be traced.

The RESUME command should not be included in a command list.

```
HPTOOLSET                              Debug Edit
ShoDebug Listing   Edit   NextKeys          Resume   Refresh   Help   MainKeys

>>DATATRACE RESULT12A
-->Stmt #28:  Data Item:  RESULT12A.  Value is "000000000000".
-->Datatrace on RESULT12A set.
>>RESUME

-->Enter Paragraph SUBTRACTION-RT of CRUN at #66.
-->Stmt #28:  Data Item:  RESULT12A.  Value is "000000010100".

-->Enter Paragraph MULTIPLICATION-RT at #77.
-->Stmt #28:  Data Item:  RESULT12A.  Value is "000000009090".

-->Enter Paragraph RESET-ALL at #90.
-->Stmt #28:  Data Item:  RESULT12A.  Value is "000000000999".

-->Enter Paragraph Z99-END at #97.
-->Stmt #28:  Data Item:  RESULT12A.  Value is "000000000000".

-->End of program.
```

Figure 4-4. Datatrace Command Screen

DATATRACE

NOTE

One can not DISPLAY, MOVE, or DATATRACE Pascal local variables or parameters that reference global types that have not been included in the main outer block compile. In other words, variables or parameters that reference newly added global types in their subprogram compile. This may be a problem with mixed languages.

DEBUGKEY FUNCTION KEY

Is available in the main function key loop while a program is running with Symbolic Debug. It replaces the Prep function key which cannot be accessed while running a program. It is a branching function key that will position you in the debug function key loop. You can return to the main loop by pressing End Run or Mainkeys.

Deletes text from an edit file. Can be reversed with the UNDO command.

Syntax

```
D[ELETE] character rangelist
```

Parameters

Character Rangelist	Defines the range(s) of characters or lines to be deleted from the current edit file Char Range [, char range] . . . where char range = line # [(column #)] [/line#[(column #)]] or ALL

Discussion

There is no Delete function key. However, the DELETE LINE or DELETE CHAR key on your terminal keyboard can be used to delete a line or character while you are in visual mode.

Example

```
>>DELETE 300/330
300
301              type
307                 mpefile  =
308                  record
309                    data     :  packed array[1..72] of char;
310                    line_no  :  packed array[1..8] of '+'..'9'
311                  end;
312
320                 srproc  =
321                  record
322                    sr
323                    procedure_name : packed array[1..30] of ' '..'Z'
324                  end;
325
>>

>>D 105
  105               lab_class = packed array[1..2]  of ' '..'Z'
>>

>>DEL 20.5(4)/22.3(5),200.1/203.1
   20.5       TYP
   20.6               lab_class = packed array[1..2] of ' '..'Z';
   20.7
   20.8               sr = integer;
   20.9
   21                 stars = packed array[1..2] of ' '..'Z';
   21.06
   21.99
   22                 fix_by = packed array[1..8] of ' '..'Z';
   22.01
   22.02              fix_version = packed array[1..8] of
   22.03
   22.1               srproc  =
   22.3             record
   201                            (*Initialize*)
   202                             Sr:=0;
   203
>>
```

DISCARD

Cancels previously entered USE commands.

Syntax

```
DISC[ARD] filename1[version designator] [,filename2[version designator]]...
```

Parameters

Filename	Is the name of a file or formal designator specified by the USE command.
Version Designator	Takes the form: #n - where n is an integer, L[ATEST] or R[EFERENCE]

Discussion

DISCARD deletes the USE entry from the workspace directory and any file equations that were generated by Toolset when the USE was issued.

DISPLAY

Is a Symbolic Debug command that causes the current contents of specified data-items to be displayed on your terminal.

Syntax

```
                                          [ O[CTAL]       ]
          {rec-item [USING t1[,t2]...] [ I[NTEGER]     ][|FOR| n |ITEMS|]}
DI[SPLAY] { data-item                  [ C[HARACTER]   ]
          {"literal" }                 [ H[EXADECIMAL]]
```

Parameters

Data-item (COBOL)

Any COBOL data-name or index-name defined in the COBOL source program. A data-item may be qualified with:

(1) OF (2) IN (3) subscript (4) periods separating the datanames where the most significant dataname is specified first.

If data-item is subscripted, the subscript must be an integer value and not a variable. If no subscript has been used, a subscript of 1 will be assumed.

You cannot reference a data-item in a program or subprogram other than the one that is currently executing.

Data-Item (Pascal)

Any Pascal variable declared in the source program. Standard Pascal selectors are used to qualify arrays and records.

Literal

Displays the delimited string. Although double quotes are not valid in standard Pascal, double quotes and single quotes are allowed as delimiters.

If a delimiter is a character in the literal, it must appear twice consecutively.

OCTAL
INTEGER
CHARACTER
HEXADECIMAL

These keyword options override the implied format of the specified data-item. If the CHARACTER option is used, the contents of the entire item are displayed as a continuous string of ASCII characters. Non-printing characters are replaced by a period.

Rec-Item

A data-item of Record type.

USING t1[,t2]...

Pascal Only

Used when you want to display a specific variant part of a rec-item. T1...Tn represent tag field constants that you may specify to explicitly select different variant parts of a record in the display.

If no tag field constant is specified in the DISPLAY command for discriminated union records, the default is to use the actual tag field

data-item to select the variant part. This default can be overriden by using the tag field options constant of this command.

If no tag field constant has been explicitly specified for free union records, the first variant of the record will be displayed by default.

FOR N ITEMS (COBOL) — Is used to display a specified number of table elements. This clause can only be used if the data item contains an OCCURS clause in its DATA DIVISION declaration. "N" is assigned the value of 1 if this clause is not specified.

FOR N ITEMS (Pascal) — Is used to display a specified number of characters in a STRING or a specified number of elements in an ARRAY. When this clause is specified for a string, the string must be subscripted. "N" characters of the string will be displayed beginning with the character indicated by the subscript.

Discussion

The DISPLAY command displays the current contents of data-items you specify on your terminal according to formats implied by the type descriptions in the Data Division of your source file (COBOL) or according to the formats implied in the declarations of your source file (Pascal).

COBOL

Data Division Formats

(1) Data with PICTURE X or A is output as ASCII characters.

(2) Picture 9 data is output in a format consistent with the COBOL Display statement.

(3) Computational-3 packed data is output as decimal digits. The sign bits are interpreted and output before the number. Leading zeros are displayed.

(4) Computational binary data is output as decimal integers. Negative signs are output preceding the number.

Table Items

If the data-item is a table item with an actual or assumed subscript (i), the [FOR n ITEMS] clause has the effect of issuing a DISPLAY command for each item in the table with subscript (i) to (i+n-1) inclusive.

Group Items

The default format for a group item is to display the entire item in character format.

DISPLAY

PASCAL

The DISPLAY command displays the current contents of specified data-items according to the formats implied in the source program's type declarations. Only those data-items that are active at the current breakpoint may be displayed.

Strings

The default display of variables with a structure type STRING is CHARACTER format. The first and last characters of the string are indicated by enclosing them in quotes. The current and maximum lengths of the string are also shown.

Record

The default display of data-items of Record type is based on its declaration in the source. This means that each field of the record is treated as its own separate data-item with its own separate default display type. You have the option of displaying the record in OCTAL, INTEGER, CHARACTER or HEXADECIMAL. In these cases, the data is displayed contiguously.

If you qualify the record with a particular field, such as MyRec.Field1, only that field will be displayed in its declared type, unless it is overriden by the OCTAL, INTEGER, CHARACTER or HEXADECIMAL option.

If the record is a union type, the fixed and a variant part of the record will be displayed.

Pointer

The default display of Pointer type variables is the OCTAL address in the pointer or NIL.

Enumerated

Variables of enumerated type are displayed using the values associated with that enumerated type.

Set

Data-items of the structured type SET are displayed as members of the base type of that SET.

BOOLEAN, REAL, LONGREAL

The default display for BOOLEAN type variables is TRUE or FALSE. The default display for variables of REAL type is real. The default display for variables of LONGREAL type is longreal.

Function or Procedure

Functions or procedures that are parameters to procedures are displayed with the name of the procedure they are parameters to along with the external program label as it appears in the Segment Transfer Table.

Character, Integer and User-Defined

Displays of variables that are of CHARACTER or INTEGER type are displayed in the corresponding default format, as are user defined variables of CHARACTER or INTEGER base type.

Constants

The display of symbols that are simple CONSTANTS contains the value and the source statement where it is declared. If the symbol is a structured Constant, including Arrays, Strings, Records and Set Constants, only the source statement will be displayed.

Withs

If execution has been suspended inside of a with construct, then fields inside the with referenced record may be displayed by simply specifying the field names.

Using Clause

When displaying a record with fields that are records, and both contain variant parts, the USING clause parameters are used to select which variant of each record to display. Consider the following code and examples:

Example

```
TYPE Case1 = (V1,V2,V3);
     Case2 = (V21,V22,V23);

VAR Rec : RECORD
            F1 :    RECORD
                      F1:   INTEGER;
                      F2:   INTEGER;
                      CASE CASE1 OF
                        V1:  (F21 : INTEGER);          {(1)}
                        V2:  (F22 : CHAR);             {(2)}
                        V3:  (F23 : RECORD             {(3)}
                                      F1 : BOOLEAN;
                                      F2 : REAL;
                                      CASE CASE2 OF
                                        V21 : (F21: INTEGER);
                                        V22 : (F22: BOOLEAN); {(3)}
                                        V23 : (F23: REAL);
                                    END;

                    END;
              F2: CHAR;
              CASE CASE1 OF
                V1 : (F21: REAL);    {(1) BY DEFAULT}
                V2 : (F22: INTEGER); {(2)}
                V3 : (F23: BOOLEAN); {(3)}
            END;
```

NOTE

One can not DISPLAY, MOVE, or DATATRACE Pascal local variables or parameters that reference global types that have not been included in the main outer block compile. In other words, variables or parameters that reference newly added global types in their subprogram compile. This may be a problem with mixed languages.

The following DISPLAY commands illustrate the sequence that Toolset uses when displaying variants within nested records. The variant displayed by each DISPLAY command is labeled with the number of that example in the sample code above.

(1) DISPLAY REC USING V1

Displays the variant V1 located in the nested record F1 and, by default, the variant V1 in the outer record REC.

```
-->Stmt #4 Var:  REC  Record
   F1 - Record
    F1  = 10
    F2  = 20
    F21 = 30
   F2  = 'a'
   F21 = 1.034000E+2
```

(2) DISPLAY REC USING V2, V2

Displays the variant V2 located in the nested record F1 and the variant V2 located in the outer record REC.

```
-->Stmt #4  Var:   REC  -  Record
   F1  -  Record
    F1  = 10
    F2  = 20
    F22 = 'z'
   F2   = 'a'
   F22  = 30
```

(3) DISPLAY REC USING V3, V22, V3

Displays the variant V3 located in the nested record F1. Since the variant V3 is of record type with its own variant, this command also displays the variant V22 located in the record F23 nested 2 deep inside REC. Lastly, this command displays the variant V3 located in the outer record REC.

```
-->Stmt #4   Var:   REC - Record
   F1   -   Record
    F1  =  10
    F2  =  20
    F23 -  Record.
     F1 =  TRUE
     F2 = -3.4577008E+2
     F22 =  FALSE
   F2 =  'A'
   F23 =  TRUE
```

The fixed fields for all of these records are also displayed.

```
HPTOOLSET                                Debug Edit
ShoDebug Listing   Edit   NextKeys          Resume   Refresh   Help   MainKeys


-->Begin execution of ONEP.MANU.TOOLS.
-->Entry is SALARY.
>>DISPLAY COUNT-FIELD
-->Stmt #17:  Data Item:  COUNT-FIELD.  Value is "0000000".
>>DISPLAY COUNT-FIELD OCTAL
-->Stmt #17:  Data Item:  COUNT-FIELD.  Value is "030060 030060 030060 0300__".
>>
```

Figure 4-5. Display Command Screen (COBOL)

Pascal

Given the following:

```
VAR Integer_Var:        INTEGER;
    Boolean_Var:        BOOLEAN;
    Char_Var:         CHAR;
    Real_Var:         REAL;
    Longreal_Var:       LONGREAL;
```

```
(1)  >>DISPLAY Integer_Var
     -->Stmt #101:    Var:    INTEGER_VAR  = 1234

     Displays the value of INTEGER_VAR.
```

```
(2)  >>DISPLAY Boolean_Var
     -->Stmt #102:   Var:   BOOLEAN_VAR = TRUE

     Displays the value of BOOLEAN_VAR.
```

Given the following:

```
TYPE Polys = (circle, square, rectangle, triangle);
     Polygon = RECORD (*fixed part and tagless variant part*)
                 Poly_Color:   (red, yellow, blue);
                 CASE Polys OF
                    circle:          (Radius:   INTEGER);
                    square:          (Side:   INTEGER);
                    rectangle:       (Length, Width: INTEGER);
                    triangle:        (Base, Height:  INTEGER);
                END;
VAR Figure:    Polygon;
```

```
(1)  >>DISPLAY Figure
     -->Stmt #76:    Var:    FIGURE - Record
        POLY_COLOR = RED
        RADIUS = 42

(2)  >>DISPLAY Figure Using Rectangle
     -->Stmt #75:   Var:  FIGURE    -    Record
        POLY_COLOR  =RED
        LENGTH = 42
        WIDTH  = 21
```

Given the following:

```
TYPE Fruit = SET OF (apple, banana, cherry, peach, pear, pineapple); VAR
Fruits : Fruit;
```

```
(1)  >>DISPLAY FRUITS
     -->Stmt #34:  Var:  FRUITS = [APPLE, BANANA, PEAR]
```

DISPLAY FUNCTION KEY

Is available when the compiler LISTING is displayed in visual mode. To use this key:

(1) Display the compile listing by pressing the Listing Key.

(2) Position the cursor at the data-item whose value you want displayed and mark the beginning and the end of the item with the MARK function key.

(3) Press the Display function key

EDIT

Opens a file for editing.

Syntax

```
ED[IT] [filename] [linepos]
```

Parameters

Filename	Name of the file you want to edit. It must be a TSAM file that is owned by the current workspace. If you do not name a file, Toolset will display the currently open edit file. If no file is open for editing, Toolset will open the file associated with the current context or prompt you for a name.
Linepos	The number of the line at which you want your edit file displayed. If you do not specify a line number, the beginning of your file will be displayed, or it will begin at the point at which you last viewed it.

Discussion

Edit Files

You can edit only one file at a time. It must be the latest version of the file and it has to belong to your current workspace. An edit file can be:

(1) a file named in a command or a file marked in the SHOW FILES or SHOW SOURCE menu

(2) a file associated with the line marked or the line at the top of your screen when a listing is displayed or

(3) a file associated with the current breakpoint when running a program with Symbolic Debug.

If the specified edit file does not exist, a new file will be created with its name. The default group and account of your file is the same as the group and account of the current workspace. They may be different from your log-on group and account.

When an edit file is created, the SET EDIT menu is displayed to allow you to change any of the defaults. The default language is the same as the workspace language.

Example

```
>>EDIT MYFILE
-->File MYFILE.group.account open for editing
```

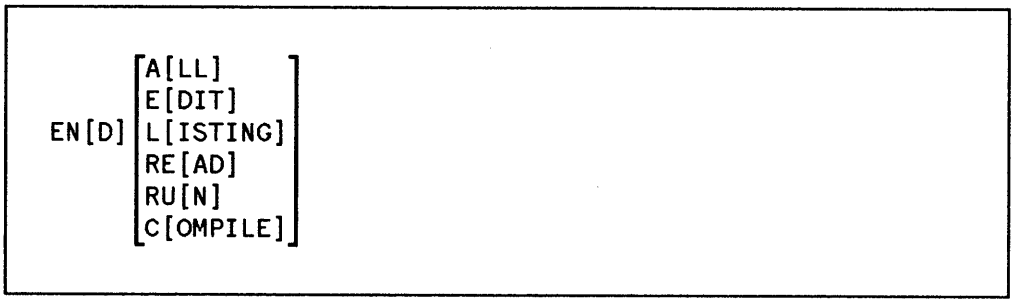
EDIT FUNCTION KEY

Is the same as entering the EDIT command without parameters. If Toolset cannot determine which file you want to edit from your current environment, you will be prompted for a filename.

END

Terminates a Toolset function or activity.

Syntax



Parameters

ALL	Terminates all ongoing Toolset activities: editing, listing, reading, running and compilation. Toolset will prompt you before terminating the compilation.
EDIT	Terminates editing of the current edit file.
LISTING	Terminates reading of the current listing file.
READ	Terminates reading of the current file.
RUN	Terminates the running of the current program file.
COMPILE	Terminates the compilation.

Discussion

When END is entered, Toolset closes all associated files. If you use the All parameter when a compile is active, you will be asked if you want to terminate the compile. When listed individually, the activities will terminate without query.

Example

```
>>END ALL

-->END OF PROGRAM.

-->EDIT FILE CLOSED.

>>
```

END EDIT FUNCTION KEY

Ends the edit function, closes the file and puts you in command mode.

END LIST FUNCTION KEY

Ends the list function, closes the file and puts you in command mode.

END MENU FUNCTION KEY

Returns you to the mode and function keys which were active before you entered the menu. If a SET menu is displayed, none of the changes will be saved.

END READ FUNCTION KEY

Ends the read function, closes the file and puts you in command mode.

END RUN FUNCTION KEY

Terminates the execution of your program. Returns you to the main function key loop.

EXIT

Exits Toolset and returns you to MPE.

Syntax

```
EX[IT]
```

Discussion

If a compile is in progress, Toolset will ask you if you want to abort the compile before it processes the EXIT. If you respond no, the EXIT process will terminate.

Example

```
>>EXIT

END OF PROGRAM

:
```

EXIT FUNCTION KEY

Exits Toolset and puts you into MPE.

Locates a specific line of text in the file you are editing or reading.

Syntax

```
F[IND] [  [ALL] find-string    [ |IN| character rangelist] ]
       [   line position                                   ]
       [level1[#line position]                             ]
```

Parameters

Find-String	Defines the text you want to locate. By default, only the first occurrence of the specified string is found. The find-string must be less than 78 characters long including delimiters. Strings must be delimited with single or double quotes or brackets. If the string is delimited with double quotes, only occurrences surrounded by one or more spaces or punctuation are found. If a string is delimited with single quotes, any occurrence of the string will be found. Strings delimited by brackets affect both upper and lower case occurrences.
Character Range List	Identifies that portion of the file which you want Toolset to search for the find-string. If the parameter is omitted, the entire file is searched. The search begins at the first line if you are in command mode or the top line on the screen if you are in visual mode.
Line Position	Defines the line you want Toolset to find.
ALL	Provides the list of all occurrences of the find-string in the character range list.
Level1#Line Position	Is allowed when you are displaying a Pascal listing. It may be necessary to specify the line position (procedure name) because each level 1 procedure is numbered beginning with 0. If the procedure name is not included, the line position will be found in the procedure last accessed.

Discussion

If no string or line number is specified, the last FIND string that you entered will be repeated. To view the last FIND string, use the SET ENVIRONMENT command.

Example

```
(1) >>FIND ALL "ABC" IN 220/260

    225  DISPLAY "EXAMPLE: ABCD    ABC     1ABC
    245  DISPLAY "EXAMPLE2:  ABCDEFGHIFKLMNOP
>>
>>FIND 300
  300              DISPLAY "THIS IS LINE 300.  YES, YOU VE FOUND IT.".
>>

>>FIND     'CAT' IN 10(30)/20(35)
  16       DISPLAY "INSTRUC AABB: ENTER WEEKLY CATALOG SALES (9999.99)"
>>

>>FIND ALL [AABB] IN 10/30
  16       DISPLAY "INSTRUC AABB : ENTER WEEKLY CATALOG SALES (9999.99)"
  18       DISPLAY "INSTRUC AABB : ENTER AMOUNT OF SALES DOLLARS (9999
>>

>>FIND INPUT_PROC#2
00002              48.000          1            END;
>>
```

FIND FUNCTION KEY

Pressing the Find function key will:

(1) Cause Toolset to prompt you for the line number or the string you want to locate or

(2) If the function key has been used previously, cause the former string to appear in parentheses. To find the next occurrence of that string, press return. If you want to find a different string or line position, type it following the prompt.

FIND -N

Finds the line that is N lines before the first line currently displayed on your screen. N is an integer value that is set by using the SET ENVIRONMENT menu or Set N function key.

FIND +N

Finds the line that is N lines after the first line currently displayed on your screen. N is an integer value that is set by using the SET ENVIRONMENT menu or SET N function key.

FIND ERR FUNCTION KEY

Displays the compile time errors and warning messages. To view your error or warning messages after you have compiled your source file, press the Listing key, press the Nextkeys key twice and then press the Find Err key.

In COBOL programs, the Find Err key positions you at the error summary at the end of your listing.

```
HPTOOLSET                          Listing Edit
 Mark    Clr Mark PrevKeys Nextkeys          Edit   Find Err Listing  End List


00273     COBOL ERRORS:
00274
00275     LINE # SEQ #   COL  ERROR  SVRTY  TEXT OF MESSAGE
00276     ------ -----   ---  -----  -----  ---------------
00277
00278     00089  009700  62    004     W    MISSING SPACE.
00279
00280     0 ERRORS, 0 QUESTIONABLE, 1 WARNING
00281
00282         DATA AREA IS %000432 WORDS.
00283         CPU TIME = 0:00:06.  WALL TIME = 0:00:36.
```

Figure 4-6. Error and Warning Message Screen (Cobol)

In Pascal programs, the compiler listing is redisplayed 5 lines before the next error message. If no errors are found in your compiler listing, the following message is displayed in the message window:

No errors in the compile listing. (640)

```
HPTOOLSET                          Listing Edit
 Mark   Clr Mark PrevKeys NextKeys          Edit   Find Err Listing  End List


00006       334.000   1      WHILE PCB_ptr <> nil do
00007       335.000   2      begin
00007       336.000   2         write (PCB_ptr^.PID:3);
00008       337.000   2         PCB_ptr := PCB_ptr^.next
00008                                                ^
00008     **** ERROR #  1                    INVALID FIELD IDENTIFIER (101)
00009       338.000   2      end;
00009       339.000   1
00009       340.000   1      writeln;
00010       341.000   1
00010       342.000   1   END;   (* Procedure Print_eligible_list *)
00000       400.000   0   $PAGE 'Procedure Print_ineligible_list'$
00000
00000    &PAGE   5  HEWLETT-PACKARD   HP32106A.01.00    PASCAL/3000   (C) HEWLET
00000
00000       401.000   0   Procedure Print_ineligible_list;
00001       404.000   0
00001    &  405.000   0   (*************************************************
00001    &  406.000   0   (*
00001    &  407.000   0   (*                    FUNCTION OF PROCEDURE
```

Figure 4-7. Error and Warning Message Screen (Pascal)

GENERATE

Generates COBOL and Pascal data declarations from information in the data dictionary.

Syntax

```
          F[ILE]    [namelist][V[AR]    [line #]            ]
GENERATE  E[LEMENT][namelist][T[YPE]   [line #]            ]]][BY Increment]
          I[MAGE]             [[ENV line #]][DATA [line #]]
          C[OMAREA]           [line #                      ]
```

Parameters

FILE

Designates that a file declaration should be generated. Declarations for any type of file supported by the dictionary (KSAM files, MPE files, Image data bases, Image data sets, Vform files or VPlus forms) can be generated.

ELEMENT

Specifies that a declaration for a single data item should be generated.

IMAGE

Generates standard elements used in Image programs. These include the status words, modes and standard name lists (";", "*;" and "@";). for COBOL. A namelist should not be specified.

VPLUS

Generates a comarea and other standard VPLUS parameters.

If this parameter is omitted, you will be asked:

-->Generate FILE, ELEMENT, IMAGE, or VPLUS declarations?

Namelist

Specifies the file or element name or names for which declarations will be generated. If you have more than one file or element name, separate them by commas. If the namelist is not included, Toolset will continue to ask for names until the question is answered with a single carriage return.

-->Filename?

-->Element name?

VAR

Determine whether var or type declarations will be done for Pascal.

TYPE

ENV
DATA

Determine whether environment and/or data division declarations will be done for COBOL. If only a line number is given, the user will be asked which should be generated.

Line #

Is the source line # at which the generated code is to be placed. If a source line already exists, the code will be placed after the existing line. If you do not specify the line numbers, you will be asked for them as follows:

-->Line for ENVIRONMENT DIVISION entry?

-->Line for DATA entry?

-->Line for TYPE entry?

-->Line for VAR entry?

One or more of these question will be asked if no line # is given in the command.

A carriage return at these prompts will cause the code to be placed at the end of the file.

BY Increment

Allows you to override the default line numbering.

Example

```
Pascal

No.1
(a)  >>GEN FILE

     -->Generate TYPE declaration or VAR declaration? T

     -->Line number for TYPE declaration? 20

     -->Dictionary Password?

     -->File name? ENGINEER  <<Image data set>>

     20  type
     21     engineer =
     22      record
     23        lab_engineer : packed array[1..8] of ' '..'Z'
     24      end
     25
     26
     27  const
     28      engineer_const       = 'engineer ';
     29      lab-engineer_const   = 'lab_engineer ';

(b)  -->Filename?  MPEFILE          <<MPE file>>

     30  type
     31     mpefile =
```

```
      32       record
      33         data     : packed array [1..72] of char;

      34         line_no : packed array [1..8] of '+'..'9'
      35       end;
      36
      37   const
      38      mpefile_const    =    'mpefile ';

(c)   -->Filename?  KSAMFILE            <<KSAM file>>

      40   type
      41      ksamfile =
      42       record
      43         line_numb      :packed array [1..6] of '+'..'9';
      44         source_line    :packed array [1..66] of char;
      45         module_name    :packed array [1..8] of char
      46     end;
      47
      48   const
      49       ksamfile_const         = 'ksamfile ';
      50


-->Filename? <cr>

>>

No. 2

>>GEN FILE FORMFILE

-->Generate TYPE declaration or VAR declaration?  TYPE

-->Line number for TYPE entry?  50

50   formfile_const  =  'formfile ';
51
52   type
52      form =
52       record
53         part_numbe : packed array[1..7] of char;
54         desc_field : packed array[1..12] of char;
55         qty        : packed array[1..4] of '+'..'9';
56         price      : packed array[1..6] of '+'..'9';
57         total_price: packed array[1..8] of '+'..'9';
58      end;
59
60
61
62   const
63      form_const  =  'form ';
64
```

```
No. 3

>>GENERATE TYPE 100

-->Generate FILE, ELEMENT, IMAGE, or VPLUS declarations?  ELEMENT

-->Element name?  FIX-BY

100  type
101     fix_by = packed array[1..8] of ' '..'Z';
102

-->Element name?  DATE-RECEIVED

103  type
104     date_received = packed array[1..6] of char;
105

-->Element name?  SR

106  type
107     sr = integer;
108


-->Element name?  <cr>

>>
```

COBOL

No. 1

This example shows the generation of two files, an MPE ASCII file and a KSAM file.

When generating COBOL declarations, you will be prompted for a prefix to be attached to data element names.

```
>>EDIT ZCOBOL;
>>GENERATE
-->Generate FILE, ELEMENT, IMAGE, or VPLUS declarations? F
-->Generate ENVIRONMENT entry, DATA entry or BOTH?  B
-->Line number for ENVIRONMENT entry?  50
-->Line number for DATA entry?  150
-->Filename?  MPEFILE

Prefix for data-items in MPEFILE > MPE-
 50      SELECT MPEFILE
 51          ASSIGN "MPEFILE      "
 52          ORGANIZATION IS SEQUENTIAL.
150      FD MPEFILE
```

```
151          RECORDING MODE IS F
152
153
154       01 MPEFILE-DATA
155          05  MPE-DATA          PIC X(72).
156          05  MPE-LINE-NO       PIC 9(5)V9(3).
157
-->File name? KSAMFILE
Prefix for data-items in KSAMFILE > KS-
  53      SELECT KSAMFILE
  54          ASSIGN "KSAMFILE"
  55          ORGANIZATION IS INDEXED
  56          ALTERNATE RECORD KEY KS-LINE-NUMB
  57          ALTERNATE RECORD KEY KS-MODULE-NAME
  58          RECORD KEY IS KS-KEY-VALUE WITH DUPLICATES
 158       FD KSAMFILE
 159          RECORDING MODE IS F
 160
 161
 162       01 KSAMFILE-DATA
 163          05  KS-LINE-NUMB       PIC 9(6).
 164          05  KS-SOURCE-LINE     PIC X(66).
 165          05  KS-MODULE-NAME     PIC X(8).
 166          05  KS-KEY-VALUE       PIC 9(6).
 167
-->Filename? <cr>
```

No. 2

This example shows the generation of an Image detail file.

```
>>GEN FILE SRPROC DATA 210

Prefix for data-items in SRPROC > SR-
210.1
210.2     01   SRPROC-DATA.
210.3          05  SR-SR                          PIC S9(9) COMP.
210.4          05  SR-PROCEDURE-NAME              PIC X(30).

210.5
210.6     01   DS-SRPROC                 PIC X(7) VALUE "SRPROC "
210.7     01   DI-SR-SR                  PIC X(3) VALUE "SR "
210.8     01   DI-SR-PROCEDURE-NAME      PIC X(15) VALUE "PROCEDURE-NAME "
210.9
>>
```

Pressing the Go key is equal to sequentially pressing the Compile, Prep and Run keys. It is a one step method of translating and executing your source file.

To use:

(1) Type any character in the box next to each file you want to compile, prep and run.

(2) Press the Go key.

If you have indicated more than one file, they will be processed in the order in which they appear on the Show Source screen. Each source file will be compiled into the workspace USL file.

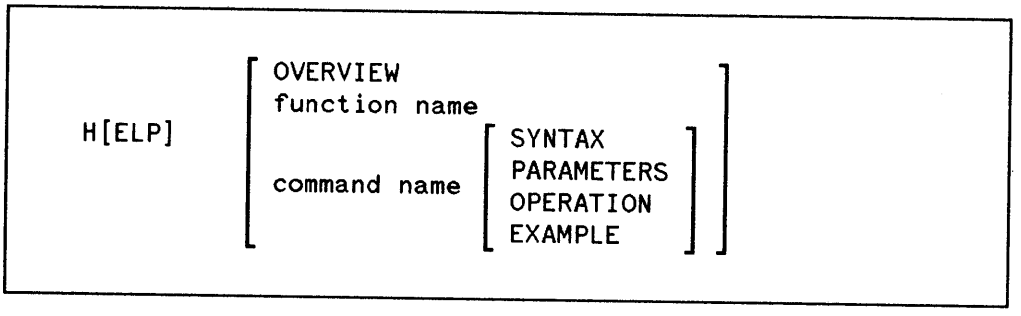
If all the compiles are processed with no errors being detected, the workspace USL file will be prepped into the workspace program file using the PREP options as they are currently defined in the SET WORKSPACE menu. If any errors are found during the compilation, the GO process will be aborted.

The RUN will start when the PREP has completed. The >> prompt will appear when the program file begins execution.

HELP

Accesses the Help Facility.

Syntax



Parameters

Overview	Provides an overview of Toolset.
Function Name	Includes softkeys, commands, Exec, workspace, Editor, Program and Debug.
Command Name	Is a Toolset command or function key.

Discussion

The Help Facility provides information on all of the key features of Toolset.

Example

```
>>Help
```

HELP FUNCTION KEY

Accesses the Toolset Help Facility.

Allows you to put a comment into a file.

Syntax

```
LA[BEL] filename[version range] "comment string"
```

Parameters

Filename	Name of the file into which the comment will be entered.
Version Range	Specifies version(s) to be labelled. Indicated by #n[/#n] or #@.
Comment String	The comment you want to put in the version(s) of a file. The string must be enclosed in quotes and <= 80 characters.

Discussion

If no version is specified, the latest version will be assumed.

Labels can be added only to files owned by the current workspace.

Example

```
>>LABEL AFILE#R "Reference Version 2/24/82"

Adds the date to the Reference version of AFILE.
```

LIBCOPY

Is used to add new modules to a copylib.

Syntax

```
LIBC[OPY] [filename] [line range] [TO modname[OF libname]]
```

Parameters

Filename	Is the name of the file from which you want Toolset to copy. If the filename is omitted, Toolset will use the existing edit file. If there isn't one, Toolset will ask you to enter the name of the file from which it will be copying.
Line Range	If no line range is given, the entire file will be copied.
TO Modname	The module name. If it is omitted, you will be asked for it.
OF Library Name	If you do not specify it, the default library file will be used.

LIBCOPY FUNCTION KEY

Is used to add new modules to a copylib.

Allows you to edit a module in an existing copylib.

Syntax

```
LIBE[DIT] [modname [OF libname] [line#]]
```

Parameters

Modname	Is the name of the module to be edited.
OF Libname	Is the name of the KSAM copy library in which the module resides. If the Libname is omitted, the workspace default library will be assumed.
Line#	Is the number of the line to be displayed at the top of the screen.

Discussion

The editing functions for the lib modules will be exactly like those in the current Toolset editor.

Only one file can be edited at a time. If an EDIT or a LIBEDIT is being done when another LIBEDIT command is entered, the previous edit file will be closed before the new one is opened.

When you do an END EDIT, Toolset will ask if you want to replace the old module. If you reply yes, the TSAM file will be copied to the copy library and then will be purged. If you reply no, the TSAM file will simply be closed. From then on, the EDIT command instead of the LIBEDIT command must be used to edit the particular TSAM file. Every time you do an END EDIT for that particular TSAM file, Toolset will ask if you want to replace the old module in the library. To avoid this query, you must change the structure from LIB MODULE to any other structure in the SET EDIT menu.

LIBEDIT FUNCTION KEY

Allows you to edit a module in an existing copylib.

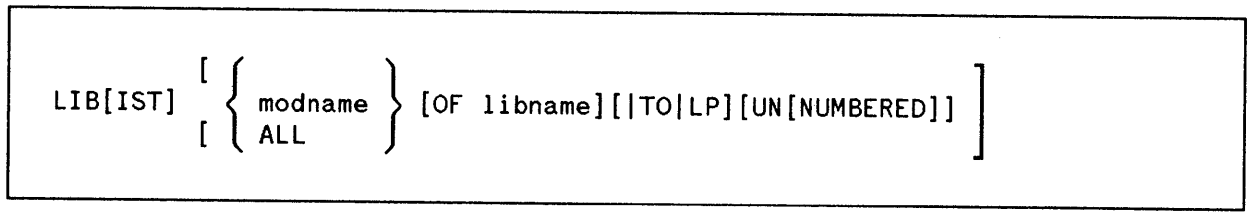
LIBKEYS FUNCTION KEY

Resides in the main function key loop and is a branching key by which the copylib editing function keys may be reached.

LIBLIST

Allows you to display the contents of a copylib without having to convert it from KSAM to TSAM format. A single module or all of the modules in the copylib may be requested.

Syntax



Parameters

Modname	If a module name is given in the LIBLIST command, only that module will be displayed.
OF Libname	Is the library name. If ALL is specified instead of a single module name, the entire library will be displayed.
TO LP	Directs the library list to TOOLLIST. If a file equation has not been done for TOOLLIST, it will default to the line printer.
UNNUMBERED	Will suppress the line numbers of the text.

Example

The following is an example of how the library list will look:

```
>>liblist all

Module1
1       source line 1
2       source line 2
3       source line 3
100     source line 100

Module2
1       source line 1
2       source line 2
3       source line 3
```

LIBLIST FUNCTION KEY

Allows you to display the contents of a copylib without having to convert it from KSAM to TSAM format. You will be prompted for the module to be listed. You may type a single module name or ALL.

LIBPURGE

Is used to delete one or more modules from a copylib.

Syntax

```
LIBP[URGE] [modname[,modname2...] [OF libname]]
```

Parameters

Modname	One or more module names may be specified. If not, you will be asked for one.
OF Libname	If the library name is omitted, the default will be used. If there is no default, you will be asked to enter the library name.

LIBPURGE FUNCTION KEY

Is used to delete one or more modules from a copylib.

Allows you to list a specific portion of your edit or read file while in command mode.

Syntax

```
L[IST] line rangelist [|TO| LP] [UN[NUMBERED]]
```

Parameters

Line Rangelist	Defines the range of lines you want listed. Pascal listings are displayed for the procedure where the file pointer is located. That location can be determined by doing a FIND *. Line range = line position [/line position] or All. If more than one range is given, separate them by commas.
TO LP	Enables you to get a line printer listing of your file. The formal file designator for the line printer is TOOLLIST. By default, the file is listed to your terminal. When in visual mode, the PRINT key can be used to list the file to the line printer.
UNNUMBERED	Lists your file without sequence numbers. The default is to list the file with numbers.

Example

```
>>LIST 100/135
 100                DISPLAY "EXAMPLE:   ABCD   ABC   1ABC".
 110                DISPLAY "EXAMPLE2: ABCDEFGHIFKLMNO  ABC".
 120                DISPLAY "MORE EXAMPLES...".
 130                DISPLAY ".....".
>>

>>LIST 15 UN
 GET-INPUT.
>>
```

LIST CHANGE

Provides a list of the changes between an early active version of a source file and a later active version of the same source file.

Syntax

```
L[IST] C[HANGE] filename[version range] [|TO| LP]
```

Parameters

Filename	Name of file whose version changes you want to list.
Version Range	Specifies the versions of the file. Changes made while each version was the Latest version will be listed. If no version is specified, changes are listed for the Latest version if you own the file and for the reference version if you do not own it or have issued a USE on it.
TO LP	Generates a line printer listing of the version changes. The output is sent to Toollist instead of to your terminal ($STDLIST).

Discussion

If a range of versions is specified, changes are listed for each version separately. If neither a version designator or a range is specified, changes will be listed for the:

(1) Latest version of an owned file

(2) Reference version of a file that you do not own or for which a USE has been done

Records that have been deleted from a version will be listed and labelled with a "D". Records that have been added will be labelled with an "A". Modified records will be labelled with an "M" followed by another line containing the modified text.

Example

```
>>LIST CHANGE YOURFILE#3/#LATEST

-->VERSION #3

A       321
A       321.01      [LINECOUNT -- COUNT LINES IN STANDARD INPUT ]
A       321.02      PROCEDURE LINECOLUNT;
A       321.03      VAR NL   :  INTEGER;
A       321.04          C   :  CHARACTER;
A       321.05      BEGIN
A       321.06       NL:=0;
A       321.07       WHILE (GETC(C) <> ENDFILE) DO
A       321.08         IF (C = NEWLINE THEN NL:= NL+1;

A       321.09         PUTDEC(NL,1);
A       321.1          PUTC(NEWLINE);
A       321.11      END;

-->VERSION #4

M       321.03      VAR NL : INTEGER;
        321.03      VAR NL : INTEGER; C : CHARACTER;
D       321.04          C : CHARACTER;

-->VERSION #5

A        61.71
M       321.09      PUTDEC(NL,1);
        321.09      PUTDEC(NEWLINE,1);
M       321.1       PUTC(NL);
        321.1       PUTC(NEWLINE);
A       322


>>PURGE YOURFILE#4
>>LIST CHANGE YOURFILE#3/#6
```

LIST CHANGE

```
-->VERSION #3

A       321
A       321.01      {LINECOUNT -- COUNT LINES IN STANDARD INPUT}
A       321.02      PROCEDURE LINECOUNT;
A       321.03      VAR NL : INTEGER;
A       321.04          C : CHARACTER;
A       321.05      BEGIN
A       321.06       NL:=0;
A       321.07       WHILE (GETC(C) <> ENDFILE)  DO
A       321.08        IF (C  =  NEWLINE)  THEN NL:= NL+1;
A       321.09        PUTDEC(NL,1);
A       321.1         PUTC(NEWLINE);
A       321.11      END;
M       360          INPUT_PROC;
        360          COPY;

-->VERSION #5

A        61.71
M       321.03      VAR NL  :  INTEGER;
        321.03      VAR NL  :  INTEGER; C : CHARACTER;
D       321.04          C : CHARACTER;
M       321.09      PUTDEC(NL,1);
        321.09      PUTDEC(NEWLINE,1);
M       321.1       PUTC(NL);
        321.1       PUTC(NEWLINE);
A       322

-->VERSION #6
```

LISTING

Displays the compile listing for the most recently compiled source file.

Syntax

```
              Program-ID
LISTI[NG] [             ] [Stmt#]
              Procedure
```

Parameters

Program-id	Identifies the particular COBOL program for which you want to obtain a listing. The listing is for the most recent compile of the program you specify. If no Program-id is specified, Toolset will determine the listing from your current operation or prompt you for a program id. This is the Program-id given in the source file and will match the one given in the identification division.
Procedure	Identifies the particular level 1 procedure in a Pascal program for which you want a listing. If no procedure is specified, Toolset will determine the listing from your current operation or prompt you for a procedure name.
stmt#	Allows you to display the listing at a particular statement in a COBOL or Pascal program.

Discussion

The LISTING command enables you to view (on-line) the listing generated from the compiled source file. Since the listing reflects the last compiled version of your source file, it may be different from the edit file which is the LATEST version.

LISTING FUNCTION KEY

Displays the compile listing generated from your most recently compiled source file.

MAINKEYS FUNCTION KEY

Is located in the Debug function key loop and in the Copylib Editing function key set. It is a branching key which returns you to the main function key loop. When you are in the main loop, DebugKey can be used to return to the Debug loop and Lib Keys will return you to the copylib editing keys.

MARK FUNCTION KEY

The Mark Function Key is available when you are displaying an edit file or listing file in visual mode. It allows you to specify character positions or line positions to be acted on by other keys such as Copy, Move, Add, At and Display.

MARKING CHARACTERS

To mark a character, position the cursor at the character you want to mark and press the Mark key once. If a string of characters needs to be marked, position the cursor at the last character and press the Mark key once again. The marked characters will be highlighted.

After you have marked the character or characters, press the appropriate key for the function you want to perform. After the function is performed, the marks will be cleared automatically.

MARKING LINES

To mark a line, position the cursor at the beginning of the line and press the Mark key twice. If you want to mark a line range, position the cursor at the last line of the range and again press the Mark key twice. And finally, press the key for the function you want to perform.

MODIFY

Allows you to change lines of text in your edit file by (D)eleting, (I)nserting or (R)eplacing characters or blocks of characters. Modify can be canceled with the UNDO command.

Syntax

```
M[ODIFY] line rangelist
```

Parameters

Line Rangelist

Defines the line or lines you want to modify.

If more than one range is specified, separate them with commas.

A line range has the form: line position [/line position] or ALL.

Discussion

When you enter the MODIFY command, the lines you specify in the rangelist will be listed by number on your terminal screen. To perform one of the MODIFY functions, place a D, I or R under the character(s) you want changed.

To cancel a previous Insert, Delete or Replace, enter a U on the same line you just modified. The first (U)ndo returns you to the previous level of modifications.

When you do not want to make any additional changes to the line, press return. To exit the modify mode, enter control-y. Your line will be left unchanged.

Example

```
    >>MODIFY 15
    15                  ABCDEFFGIJK

                              D IH

    15                  ABCDEFGHIJK
```

Displays line 15 of your edit file and (d)eletes the second F. It then (i)nserts an 'H' between G and I.

```
>>MODIFY 8
8                   12445678
                      R3

8                   12345678
```

Displays line 8 of your edit file and (r)eplaces the first 4 with a 3.

MOVE (EDIT)

Moves text from one location to another. Can be cancelled with the UNDO command.

Syntax

```
MOV[E] character range |TO| character position [BY increment]
```

Parameters

Character Range	Defines the range of source characters and/or lines you want to move. Line position [(column #)] [/line position [(column #)]] or ALL
Character Position	Is the location where the character range is to be moved. Line position [(column #)]
BY Increment	Defines the increment for moving text. If no increment is specified, it will be the default increment, implied increment or space available.

Discussion

After the MOVE command is executed, the moved text will exist only at the new location.

When moving a string of characters into a line which results in the need for more than one record, the line will be split where the insertion is made. Characters will be inserted until the record (line) is full. When this occurs, a new record will be started.

Example

(1) >>MOVE 37/60 TO 85 BY .1

```
Moves lines 37 through 60 to line
85 with an increment of .1.
```

(2) >>MOVE 12.1(20)/12.3(20) to 200

```
Moves all the text from column 20 of line 12.1
through column 20 of line 12.3 to line 200.
```

NOTE

One can not DISPLAY, MOVE, or DATATRACE Pascal local variables or parameters that reference global types that have not been included in the main outer block compile. In other words, variables or parameters that reference newly added global types in their subprograms compile. This may be a problem with mixed languages.

MOVE FUNCTION KEY

To use the Move Function key:

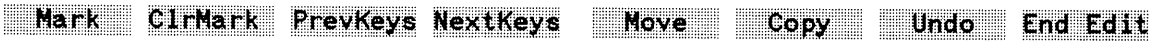
(1) Display the text you want to move.

(2) Mark the first and last characters or lines you want to move with the Mark function key (2 marks for lines and one mark for characters).

(3) Move the cursor to the destination line and mark it.

(4) Press the Move key.

Edit Move Function Key Set



MOVE (SYMBOLIC DEBUG)

Transfers the value of a literal, figurative constant or data-item to another data-item.

Syntax

```
        {literal     }
MOV[E] {data-item-1 } TO data-item-2 [|FOR| n |ITEMS| ]
        {fig-constant}
```

Parameters

Literal (COBOL)

Is a character string that defines itself rather than representing some other value. The value being transferred may be numeric or non-numeric, signed or unsigned. Non-numeric values require surrounding double quotes.

This is a sending field.

Literal (Pascal)

May be a signed or unsigned numeric literal, a non-numeric literal or a simple constant identifier. Although double quotes are not valid in standard Pascal, Toolset allows double quotes as well as single quotes. If a delimiter is a character in the literal, it must appear twice.

Data-Item 1(COBOL)

The sending field for the MOVE command. A data-item is any COBOL data-item or index-name defined in your source program. If subscripted, the subscript must be an integer value. If the data-item is a table item and unsubscripted, the subscript 1 is assumed. Level 88 items are not allowed.

Data-Item (Pascal)

If data-item 1 is an array item with an actual or assumed subscript (j), the subscript of data-item 1 takes values (j) to (j+n-1) inclusive when functioning as the source item for the MOVE command.

If a subscript is used, it must be an integer, character or identifier that is part of an enumerated type, depending on the type of the array subscript.

Figurative Constant (COBOL)

A value that has been assigned a fixed data-name in COBOL. ZEROS and SPACES are the only figurative constants allowed in Toolset.

Figurative Constants (Pascal)

Those standard constants recognized by the Pascal compiler: FALSE, TRUE, MININT, MAXINT and NIL.

Data-item 2

The receiving field.

For N Items (COBOL)

Number of times data-item 1 is to be moved. N is a positive or unsigned integer. This clause can be used if data-item 2 contains an occurs clause in its Data Division declaration. A value of 1 is always assumed for n if this is not specified.

The Lower Bound of a table is always assumed to be one. The Upper Bound is always assumed to be the maximum value regardless of any possible DEPENDING ON clause.

For N Items (Pascal)

This clause can be used only if data-item 2 is of type ARRAY. If data-item 2 is a one dimensional array with an actual subscript (i), using this clause causes a MOVE command to be issued for each item of the array with subscript (i) to (i+n-1) inclusive.

Specifying this parameter when data-item 1 and data-item 2 are multi-dimensional arrays and subscripts are omitted causes rows or sub-arrays to be moved.

Discussion

COBOL

The Symbolic Debug MOVE command allows you to transfer the value of a literal, figurative constant or data-item to another data-item. When the sending field or literal is shorter than the receiving field, data-item 2 is padded with zeros and spaces as required. If data-item 2 is shorter, the value is truncated. The permissible types for data-item 1 and data-item 2 and conversion are defined according to the COBOL MOVE statement.

Subscripts

If data-item 2 is a table item with actual or assumed subscript (i), the FOR N ITEMS clause has the same effect as issuing a MOVE command for each item of the table with subscript (i) to (i+n-1) inclusive.

If, in addition, data-item 1 is a table item with actual or assumed subscript (j), the subscript of data-item 1 takes the values (j) to (j+n-1) inclusive when functioning as the source item for the MOVE command.

Symbolic Debug aligns decimal points (".") in the sending and receiving fields for you.

Pascal

The MOVE command transfers the value of a literal, constant or data-item to another data-item. Data-item 1 and Data-item 2 subscripts can be integers, characters or identifiers that are part of an enumerated type, depending on the type of array subscript.

The following are rules for moving string literals or variables of type STRING, packed array of CHAR (PAC) or CHAR:

String Variable

If data-item 2 is a STRING variable, the sending field must be a STRING variable or a STRING literal whose length is less than or equal to the maximum length of data-item 2. The sending field cannot be a PAC or CHAR variable. The MOVE command sets the current length of data-item 2.

PAC Variable

If data-item 2 is a PAC variable, the sending field must be a PAC of equal length or a STRING literal whose length is less than or equal to the length of data-item 2. If the sending field is a STRING literal that is shorter than data-item 2, then data-item 2 is padded with blanks.

CHAR Variable

If data-item 2 is a CHAR variable, the sending field can be a CHAR variable, a STRING literal of length 1 or a subscripted string variable.

Example

```
COBOL

(1)  Figurative Constant

>>MOVE ZERO TO TABLE(5) FOR 20

Moves the value ZERO starting at element 5 of TABLE for 20 times.


(2)  Data-Item

>>MOVE KOUNT TO TOT-KOUNT

Moves the value of data-item KOUNT to the data-item
TOT-KOUNT.

(3)  Numeric Literal

>>MOVE 25 TO KOUNT

Moves the value 25 to the data-item KOUNT which is defined as
a picture 99.

(4)  Table Item

>>MOVE MY-TABLE TO THEIR-TABLE(2,4,3) FOR 5

Moves the items in MY-TABLE to THEIR-TABLE in the
the following manner:


MY-TABLE        (1,1)     (1,2)     (1,3)     (2,1)     (2,2)

THEIR-TABLE     (2,4,3)   (2,4,4)   (2,4,5)   (2,5,1)   (2,5,2)


MY-TABLE is defined as a (3*3) dimension table and THEIR-TABLE
is defined as a (5*5*5) dimension table.

Since MY-TABLE has no explicit subscript, it defaults to the
first element in the table.  The subscript that is nested the
deepest varies the fastest.
```

MOVE (SYMBOLIC DEBUG)

Pascal

Moving Arrays

(1) >>MOVE AR1 TO AR2

Moves all elements of ARRAY AR1 to ARRAY AR2.

(2) >>MOVE AR1[1] TO AR2[5]

Moves row 1 of ARRAY AR1 to row 5 of ARRAY AR2.

Moving Data Items

Given the following:

```
TYPE Polys = (circle, square, rectangle, triangle);
     Polygon = RECORD (*fixed part and tagless variant part*)
                 Poly_Color:  (red, yellow, blue);
                 CASE Polys OF
                    circle:      (Radius:  INTEGER);
                    square:      (Side:    INTEGER);
                    rectangle:   (Length, Width:   INTEGER);
                    triangle:    (Base, Height:    INTEGER);
               END;
VAR Figure:    Polygon;
```

(1) >>MOVE BLUE TO FIGURE.POLY-COLOR

-->STMT #76: Var FIGURE.POLY-COLOR=BLUE

Moves the value Blue into the Record Polygon.

```
Moving Numeric Literals
Given the following:

VAR Integer_Var:   INTEGER;
    Boolean_Var:   BOOLEAN;
    Char_Var:  CHAR;
    Real_Var:  REAL;
    Longreal_Var:  LONGREAL;

(1)  >>MOVE 98.456601E-22 to Real_Var
     -->Stmt #104:  Var:  REAL_VAR = 9.8456601E-21

     Moves the value 9.8456601E-21 to the data-item REAL_VAR.


Moving Characters and Literal Strings


Given the following:

VAR Char_Var:    CHAR;
    Str1:  STRING[80];


(1)  >>MOVE 'Z' TO CHAR_VAR
     -->Stmt #400:  Var:  CHAR_VAR  = 'Z'

     Moves the character Z to the variable CHAR_VAR.

(2)  >>MOVE "This is the good life."  TO STR1
     -->Stmt#401:  Var:  STR1 (22/80) =  This is the good life."

     Moves the literal string above to the variable STR1 and sets
     the current length of STR1 to 22 characters.
```

NOTE

One can not DISPLAY, MOVE, or DATATRACE Pascal local variables or parameters that reference global types that have not been included in the main outer block compile. In other words, variables or parameters that reference newly added global types in their subprogram compile. This may be a problem with mixed languages.

NEXTKEYS FUNCTION KEY

Is used to change to the next set of function keys within a function key loop. It is available on f4 at all times except when there is only one set of keys included in the loop.

NEXTPAGE FUNCTION KEY

Is available in several different contexts. In visual mode, it displays the next page of text. The size of the page is defined in the PAGESIZE field of the Set Environment menu. Default size is 20 lines. In the Show Files and Show Source menu, the NextPage key displays the remainder of the file names if they did not all fit on a single screen. It is used in the Cmd List menu to display the Cmd List for the next breakpoint. In Help, it displays the next page in the Help Facility.

Prepares a USL file and creates a Program file.

Syntax

```
PREP [uslfile, progfile]

       [;ZERODB]
       [;PMAP]
       [;MAXDATA=segsize]
       [;STACK=stacksize]
       [;DL=dlsize]
       [;CAP=caplist]
       [;RL=filename]
       [;PATCH=patchsize]
       [;NOSYM]
       [;FPMAP] [;NOFPMAP]
```

Parameters

Usl File	Is the name of the USL file to be prepared. If this parameter is omitted, the workspace USL file will be assumed.
Progfile	Is the program file into which the executable code is placed. If omitted, the workspace program file is assumed.
NOSYM	Specifies that Symbolic Debug information be omitted from the program file. You cannot RUN with Symbolic Debug if this is specified.
FPMAP	Includes symbolic PMAP information in the program file. This is the default if the program was compiled with the SYMDEBUG option. No PMAP information is included in the program file if the file is not compiled with the SYMDEBUG option.

Discussion

The TOOLSET PREP command prepares a USL file and translates the source compilation into a program file. If you do not specify a USL and a program file name, the workspace USL and program file names as defined in the Set Workspace menu will be used.

Similarly, if you do not specify any parameters with the PREP command, the PREP options defined in the Set Workspace menu will be used. For parameters that were not defined in the menu, Toolset will use the defaults of the MPE PREP command.

A workspace program file is automatically saved when it is created with the PREP command.

Example

```
>>PREP; CAP=IA,BA,PH

Prepares the workspace USL file into the workspace program file.
using the MPE default options except cap=ia,ba,ph.

>>PREP THISUSL, THISPROG
Prepares thisusl and translates it into thisprog, using
the Prep options as they were defined in the set workspace menu.

>>PREP

Entering the PREP command without parameters is the same as
using the Prep key.  It prepares the workspace USL file and
translates it into the workspace program file using the PREP
options as defined in the set workspace menu.
```

PREP FUNCTION KEY

Prepares your workspace USL file using the PREP options that were defined in the set workspace menu.

Main Program Function Key Set

The Prep key is available in the main loop except when running a symbolic debug program.

```
HPTOOLSET                          Main Program
ShoSourc Listing  PrevKeys NextKeys        Compile   Prep    Run    Exit


>>PREP

HP32050A.01.07j SEGMENTER/3000 (C) HEWLETT-PACKARD CO 1982
--
Process times to this point are:
     CPU TIME = 2.684        SEC.
    WALL TIME = 13.672       SEC.

-
-->End of program.
>>
```

Figure 4-8. Program File Preparation

NOTE

Commands cannot be entered while the Segmenter is being accessed. You may continue when the >> prompt returns to your screen.

PREVKEYS FUNCTION KEY

May be used to move to the previous set of function keys within a function key loop. Is available on f3 in all loops that have more than two function key sets included in them.

PREVPAGE FUNCTION KEY

Is available in several different contexts. In visual mode, it displays the previous page of text. The size of a page is defined in the PAGESIZE field of the set environment menu. The default size is 20 lines. In the Show Files, Show Source and Command List menus, PrevPage is used to return to the previous page after the NextPage function key has been used.

Sets and resets the toggle for a hard copy of the listfile during compile.

Syntax

```
PRI[NT] {ON }
        {OFF}
```

Parameters

ON	Turns on the PRINT toggle to enable hard copy listings to be generated when compiles are executed.
OFF	Turns off the PRINT toggle.

Discussion

The PRINT command is used to set and reset a toggle which determines whether a hard copy of the listfile will be produced during compiles. When the toggle is on, a hard copy of the listfile is directed to TOOLLIST.

PRINT FUNCTION KEY

Is available while in visual mode. Pressing this key generates a hard copy of the currently displayed listing or source file. The listing is directed to the line printer unless an MPE :FILE command has been executed for the formal designator TOOLLIST.

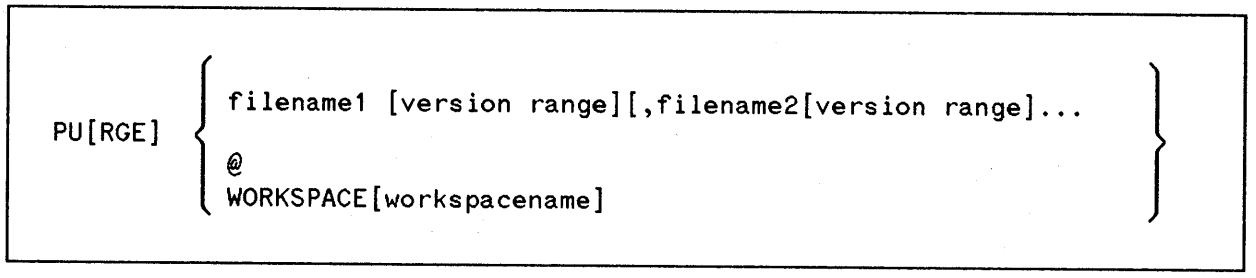
Edit Print Function Key Set

Set Edit | Print | PrevKeys | NextKeys | Cmd Mode | Refresh | Undo | End Edit

PURGE

Purges files that are owned by the current workspace.

Syntax



Parameters

Filename	Name of the file you want to purge. If there is more than one file, they should be separated by commas.
Version Range	Takes the form: #n[/#n] or #@ where n is an integer, L[atest], or R[eference]. @ = all versions.
@	Purges all owned source files in the current workspace.
WORKSPACE	Indicates that an entire workspace is to be purged. All versions of owned source files, all listing files, the USL file, program file, the KSAM listing directory and workspace directory of the current or specified workspace are purged. You will be prompted to confirm the purge if this parameter is used.
Workspacename	Allows you to specify the particular workspace you want to purge. If you do not specify a workspacename, the current workspace will be purged. Do not give this parameter if the workspace you are purging is your current workspace.

Discussion

If any active and open workspace is accessing the file you want to purge, you must wait until it closes the file. When a version is purged, that version is, of course, unavailable for future referencing. If you reference a purged version of a file, you will receive an error message. Later versions of a file are unaffected by the purging of earlier versions. If the latest version is purged, the records added in it are deleted.

The number of active (non-purged) versions of a file may be different from the highest version number because some of the earlier versions may have been purged. For example, if there were originally 10 versions of a file, and version 2, 4, and 5 have been purged, there are 7 active versions, though the latest version is #10.

Example

(1) >>PURGE MYFILE#3

Purges version 3 of MYFILE.

(2) >>PURGE MYFILE#@

Purges all versions of MYFILE and deletes it from the workspace directory. It is the same as not specifying any versions.

(3) >>PURGE @

-->Purge all owned files? YES

Purges all owned files in the current workspace except the USL file and the program file. You will be prompted to verify the purge before the files are actually deleted.

(4) >>PURGE WORKSPACE WW

-->Purge workspace WW?

All owned files, the USL and program file for Workspace WW are purged. File equations and directory listings are also deleted.

PURGE FUNCTION KEY

Purges files that are owned by the current workspace.

One or more files can be purged by entering any character in the box(es) beside the file name(s) and pressing the Purge key. When this key is pressed, you will be positioned in command mode and prompted to verify that those files should be purged.

READ

Allows a Toolset formatted file to be read without being open for editing.

Syntax

```
REA[D] [filename[version designator]] [lineposition]
```

Parameters

Filename	Name of the file you want to read. If no filename is given, Toolset will display the file that is currently open. If no filename is given and no file is open, you will be prompted for a name.
Version Designator	Allows you to indicate which version of the file you want to read. If no designator is indicated, the default version will be displayed. The default version is the LATEST version if your workspace owns the file and the REFERENCE version if it does not own the file or has issued a USE on the file. Has the form #n or #LATEST or #REFERENCE.
Line position	Allows you to indicate a particular line at which to start reading. If no line number is indicated, the first part of the file will be displayed.

READ FUNCTION KEY

Is the same as the READ command without any parameters specified.

The Read function key can be found in the Main Edit Function Key Set in the main loop or while in the Show Files Menu. If you are in the Show Files Menu, enter a character in the box beside the file you want to read. When you press the Read key, your file will be displayed on your terminal screen.

RECOVER

Recovers all versions of a file.

Syntax

```
REC[OVER] [filename[,U[NLOCK]]]
```

Parameters

Filename	Name of the file you want to recover. If it is not specified, the current Edit file will be recovered.
UNLOCK	Is an option you may use for quick recovery if you were not making modifications to your file when the abnormal termination occurred.

Discussion

The Recover command may be needed after a system failure or other abnormal termination of Toolset. This command requires that you have exclusive access to the file being recovered.

During recovery, Toolset creates a permanent scratch file. It renames this scratch file to the filename you have specified. If an error occurs during the renaming process, the file will be in your current workspace group and account.

There are some conditions under which a file may not be recoverable, such as an unreadable TSAM file directory. If this occurs, you will be advised to reload an old copy of the file.

Example

```
>>RECOVER Myfile

-->Recovery complete.
>>
```

REDO

Allows you to correct and re-execute the last command or command list.

Syntax

```
RED[O]
```

Discussion

When REDO is entered, the last command is displayed. At that time, characters can be inserted, deleted or replaced by entering I, D or R on the line below the command. Inserting and replacing must be followed by the characters you wish to insert or replace.

Editing functions can be cancelled by entering one U after the function. To return to the original command, enter two consecutive U's. If neither an I, D, R or U is entered following the REDO command, a Replace is performed by default.

Example

```
>>EDT

***Undefined Toolset command keyword.  (101)

>>REDO

EDT
  II
EDIT

<<CR>>

-->Name of file to be edited?
```

Will redisplay your screen in the event that it is accidentally destroyed.

Syntax

```
REF[RESH]
```

REFRESH FUNCTION KEY

Is the same as the REFRESH command. There is a Refresh function key available in all function key loops and within each menu. If your screen was accidentally destroyed while in one of the function key loops, pressing the NextKeys key will restore the function key labels so you can find the Refresh key.

RENAME

Allows you to change the file identification within a system.

Syntax

```
RENA[ME] filename1|,|filename2
```

Parameters

Filename1	Is the original name of the file.
Filename2	Is the new name of the file.

Discussion

RENAME can only be used within the Owner workspace. It is important to rename TSAM files while you are in Toolset instead of using the MPE :RENAME command so the name change will be recorded in the workspace directory.

Example

```
>>RENAME OLDFILE NEWFILE

  Renames OLDFILE to NEWFILE.
```

Renumbers all lines of text in your current edit file.

Syntax

```
RENU[MBER] [BY increment]
```

Parameters

BY Increment	Defines the increment by which you want to renumber your file.

Discussion

You can only renumber files that contain one version. If a file contains more than one version, it must be copied into a one version file before it can be renumbered.

If a break or crash occurs during the renumbering of a file, Toolset must recover the file before it can be used again.

Example

```
>>RENUMBER BY .01

Renumbers your edit file by increments of .01.
```

RESTORE

Restores all owned files, the USL file, program file, listing files, KSAM listing directory, and directory file information for the workspace you specify.

Syntax

```
REST[ORE] workspacename [KEEP]
```

Parameters

Workspacename	Name of workspace to be restored.
KEEP	Prevents files which already exist on the system from being restored. The workspace directory file is always restored, whether or not a KEEP is specified.

Discussion

Files must be restored into the group and account in which the workspace was created.

Restoring requires that you answer two tape mount requests at the system console. The first request restores the workspace directory and the second restores the workspace owned files, the USL file, program file and listing files, and KSAM listing directory.

Example

```
>>RESTORE WXYZ

Restores all stored files for the Workspace WXYZ, purging
any of WXYZ's files that currently exist on the system.
```

Is a Symbolic Debug command that starts or restarts program execution.

Syntax

```
RES[UME] [para-name     ]
         [section-name  ]
```

Parameters

Para-name	References a paragraph name where you want program execution to continue.
Section-name	References a section in your program where you want execution to continue.

Discussion

The RESUME command starts or restarts the execution of your program at either the location at which it was halted or a specified paragraph or section. If you specify a paragraph or section other than the location at which your program was halted, the validity of the user stack cannot be guaranteed.

Execution of Pascal programs can only be resumed at the current location.

Example

```
>>RESUME

Resumes program execution at the current location.

>>RESUME MAIN-LOGIC

Causes program execution to restart at paragraph MAIN-LOGIC.

>>RESUME SECT-03

Causes program execution to restart at Section SECT-03.
```

RESUME FUNCTION KEY

Is the same as entering the RESUME command without parameters. Execution of the program will restart at the current execution location.

The Resume function key is available in the Debug Edit, Debug Utility and Listing Debug Function Key Sets.

Debug Edit Function Key Set

```
ShoDebug Listing   Edit   NextKeys   Resume  Refresh    Help   MainKeys
```

Debug Utility Function Key Set

```
 Trace   Retrace   Calls   NextKeys   Resume  At Next   Clear   End Run
```

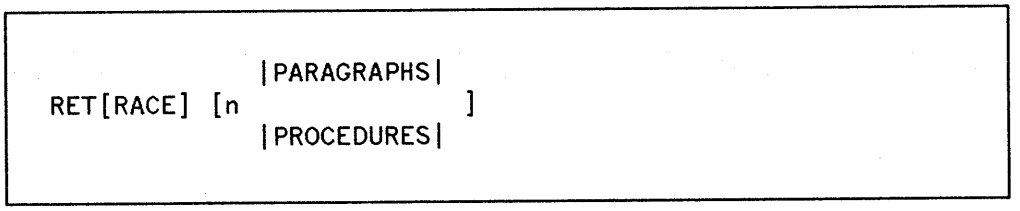
Listing Debug Function Key Set

```
  Mark   Clr Mark PrevKeys NextKeys   Resume     At     Display  End List
```

RETRACE

Lists the most recently executed paragraph(s) or procedure(s) name(s).

Syntax



Parameters

N	The number of paragraphs or procedures to list. Must be less than or equal to 10. The default is 1.

Discussion

The RETRACE command lists the last n paragraphs or sections that have executed in a COBOL program ending with the most recent one. For Pascal programs this command provides a list of the last n procedures entered ending with the most recent one. The RETRACE display is identical to that of the TRACE command.

Example

```
HPTOOLSET                    Debug Utility
Trace  Retrace  Calls  NextKeys        Resume  At Next  Clear  End Run


>>AT HANDLE_INSTRUCTION
-->Breakpoint set.
>>RESUME

-->Breakpoint in HANDLE_INSTRUCTION.
-->Statement Number is #31.

>>RETRACE 10
  -->      Enter SKIP_BLANKS at #27
  -->      Enter CONVERT_NUMBER at #15
  -->      Enter SKIP_BLANKS at #27
  -->      Enter CONVERT_NUMBER at #15
  -->      Enter SKIP_BLANKS at #27
  -->      Enter CONVERT_NUMBER at #15
  -->Enter A at #7
  -->Enter LOCATE_PCB at #4
  -->Enter PRINT_SEGMENT_TABLE at #4
  -->Enter HANDLE_INSTRUCTION at #31
>>
```

Figure 4-9. Retrace Command Screen

This example displays the last 10 paragraphs that executed in the program before the RETRACE command was entered.

RETRACE FUNCTION KEY

Executes a retrace using the default (1) number of paragraphs and procedures.

Debug Utility Function Key Set

```
Trace  Retrace  Calls  NextKeys  Resume  At Next  Clear  End Run
```

Executes a program file.

Syntax

```
RUN [progfile] [,entrypoint]

     [;NOPRIV]
     [;LMAP]
     [;DEBUG]
     [;MAXDATA=segsize]
     [;PARM=parameternum]
     [;STACK=stacksize]
     [;DL=dlsize]

           G
     [;LIB={P}]
           S

     [;NOCB]
     [;INFO=string]

     [;STDIN = [*formaldesignator] ] [           [*formaldesignator] ]
               [fileref            ] ] [;STDLIST = [fileref [,NEW    ] ]
               [$NULL              ] ]             [$NULL            ] ]
```

Parameters

Progfile	The name of the program file. If the file is not fully qualified, Toolset assumes the logon group and account, unless a USE has been done or it is the workspace program file.
Entrypoint	Point in program where execution is to begin. See the MPE Commands Manual for a description of the run options. If no run options are specified when running the workspace program, the options defined in the Set Workspace menu will be used.

Discussion

Executes the designated program file. If no program file is specified with the progfile parameter, the program file defined in the Set Workspace menu for the current workspace will be executed. If you are running the workspace program file, execution will begin under Symbolic Debug unless the file was prepared with the NOSYM option or compilation did not include the compiler control option SYMDEBUG in the source.

Symbolic Debug cannot be used on program files which are not known to the workspace. If you want to use Symbolic Debug on a program file other than the workspace program file, issue a USE command. The

workspace considers the shared file part of your workspace until a DISCARD command is performed on that file.

Function keys are deactivated while your program file is executing. When control returns to Toolset, either through termination of execution or encountering a symbolic debug interrupt, the keys will be reactivated.

RUN FUNCTION KEY

Pressing the Run key causes the program in the current workspace to execute. It is the same as entering the RUN command without parameters.

Program Function Key Set

Shosourc Listing PrevKeys NextKeys Compile Prep Run Exit

The screen below shows the program file MINEP at the entry point. In this case, Toolset has control and you can perform various Symbolic Debug functions including resuming execution of your program.

```
 HPTOOLSET                          Debug Edit
ShoDebug Listing  Edit  NextKeys        Resume Refresh  Help  MainKeys


>>RUN
-->Begin execution of ONEP.MANU.TOOLS.
-->Entry is SALARY.
>>
```

Figure 4-10. Program Execution Screen

Allows you to view and change editing options. The defaults on this menu are determined by the default workspace language.

Syntax

```
SET ED[IT]
```

```
HPTOOLSET                          Set
          Set OK                              Refresh          End Menu

   SET options for file PROJ2.MANU.TOOLS

   * SOURCE LANGUAGE ..................... [PASCAL] (COBOL, PASCAL, OTHER)
   * PROGRAM STRUCTURE ................... [MAIN   ] (MAIN, SUBPROGRAM,
                                                      INCLUDE, LIB MODULE,
                                                      OTHER)
   * PROGRAM-ID .. [                                ] (FOR COBOL MAIN OR SUBP)
   * LINELENGTH ................................ [72 ]
   * FILESIZE ............................... [3000 ]
   * GUIDE ..................................... [OFF] (ON, OFF)
   * INCREMENT ................................ [1  ]
   * TABS ..................................... [LANGUAGE] (LANGUAGE, OFF, OTHER)
                                                      (IF OTHER, ENTER BELOW)

    123456789^123456789^123456789^123456789^123456789^123456789^123456789^123
   [  *  *  *  *  *  *  *  *  *  *  *                                        ]
```

Figure 4-11. Sample Set Edit Menu for Pascal File

Example

Figure 4-11 shows an example of the SET EDIT menu. The following text describes the fields in that menu.

Source Language

Indicates which language is being used in your source program. This selection affects other Editor and Program key options, such as the format of sequence numbers. For COBOL files, the sequence number can have up to 4 digits on the left side of the decimal point and two digits on the right. For other language files, including Pascal, 5 digits can be used to the left of the decimal point and 3 to the right.

Structure

Tells Toolset the source language structure contained in your edit file. You can change this option at any time.

SET EDIT

Linelength	Sets the lenght of the logical records in your file. This should be set to the maximum number of characters you want per record, excluding the sequence number. The default is 74 for COBOL and 72 for other languages. You cannot reset this option.
Filesize	Is the number of logical records in your file. The default is 6000 records. This option cannot be reset.
Guide	Setting the Guide option on places a column guide across the first line of the screen during the visual mode editing, starting with column 7 for COBOL files and column 1 for other files, including Pascal. The guide remains there until you set the guide option off or until you end editing the file. The default is set off.
Increment	Is the default increment for your file. Is used if no increment is explicitly given, if the implied increment is larger and if a sufficient amount of space is available. An incrment is implied by the destination location you give when moving text. A text move to line 10.03 implies an increment of .01. A text move to 5.2 implies an increment of .1.
Tabs	To use the default tabs for a particular source language, set the Tabs option to other and indicate the tabs you want by entering the characters in the specific columns under the column guide on the menu. To eliminate all tabs, set tabs to off. A maximum of 10 tab locations may be set.

SET EDIT FUNCTION KEY

Is the same as the SET EDIT command without parameters.

The SET EDIT menu is displayed when a new edit file is created and when the SET EDIT command or function key is used. Any of the highlighted fields may be changed. When you are satisfied with the menu, press the Set Ok function key. If you press the End Menu key, none of the changes will be saved and the new edit file will not be created.

Function Key Set

Edit Print Function Key Set

Set Edit Print PrevKeys Nextkeys Cmd Mode Refresh Undo End Edit

SET ENVIRONMENT

Allows you to change screen oriented functions such as pagesize.

Syntax

```
SET EN[VIRONMENT]
```

Parameters

Echo	If Echo is set on (default), commands executed from the XEQ files and command lists will be echoed on your terminal screen. If Echo is set off, commands from these input sources will not echo back.
Pagesize	Is the number of lines of the EDIT and READ file that appear on your screen at one time while in visual mode. The default is 20 lines. The integer specified must be between 2 and 40 inclusive.
N	Is the value for N in the Editor function keys Find+N and Find-N. These keys allow you to go forward or backward in an edit, read or listing file by the number of lines defined by N. N is set to 10 lines by default.
Editmode	If set to command, you stay in command mode when you enter the LISTING, READ or EDIT commands. If edit mode is set to visual, you will be put into visual mode when the LISTING, READ or EDIT commands are entered, unless they are entered from a command list or XEQ file. In these instances, you will enter command mode. Visual mode is the default.
Quiet	If Quiet is set to off, you will receive a listing of the lines affected when the editor operations are performed on the source file. If Quiet is set to on, the lines affected by the editor operations such as Delete will not be listed.
Find String	Is a string to be used by the FIND command.
Change String	Defines the string to which the Find String will be changed.

Discussion

The Set Environment options remain in effect as long as you remain in Toolset. The options are not saved in your workspace. You must set them to the desired values each time you run Toolset. Any of the highlighted fields may be changed. When you are satisfied with the menu, press the Set Ok key. If you press the End Menu key, none of your changes will be saved.

Separate commands are also available for changing any of the SET ENVIRONMENT fields. For instance, you could type SET PAGESIZE=40 while in command mode and avoid displaying the menu.

SET ENVIRONMENT

SET ENVIRONMENT FUNCTION KEY

Is the same as the SET ENVIRONMENT command without parameters.

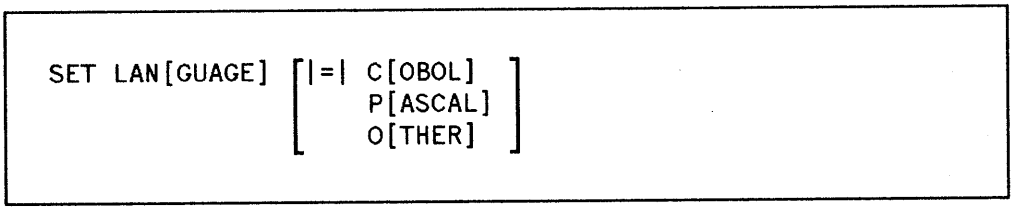
Workspace Function Key Set

Workspace Function Key Set

SET LANGUAGE

Changes the workspace default language.

Syntax



Parameters

C[OBOL]
P[ASCAL]
O[ther]

Is the source language for the program. This option cannot be reset after the creation of the file.

SET LIB

Defines a default copylib to be used with the Lib Editing commands.

Syntax

```
SET LIB[RARY] [|=| libname]
```

Parameters

Libname	Is the library name.

Discussion

If a Set Lib has been done within the workspace, the library name will be saved in the workspace root file. If a copylib with the specified libname does not exist, a KSAM file will be built with that name. Before building a new copylib, Toolset will ask for the:

-->Maximum size for libname?

You can display information about the workspace default copylib by executing the SHOW LIB command.

SET LIB FUNCTION KEY

Is the same as the SET LIB command without the parameter.

Designates the Reference version of a source file.

Syntax

```
SETR[EF] filename1[version designator1][,filename2[version designator2]]
```

Parameters

Filename	Name of the file for which you want to specify a reference version.
Version Designator	Specifies the particular version of the file. Takes the form: #n where n is an integer, L[atest] or R[eference]

Discussion

Only the owner of a file can set the reference version for other users and setting it requires exclusive access to the file.

You can display the reference versions of files with the SHOW FILES command.

Example

```
The current LATEST version of AFILE is version #5.  If the
following command instructions are performed on AFILE, the reference
version will no longer be the LATEST version.

(1)  >>SETREF AFILE#3

     Sets the reference version to #3.  The users reference
     version #3.  The owner of AFILE references the LATEST
     version, version #5, by default.

(2)  >>SETREF AFILE

     Sets the reference version equal to the LATEST, version #5.
     Users and the file owner both reference version #5 by
     default.

(3)  >>SETVERSION AFILE

     Freezes version #5 so that no further changes can be made
     to it.  Sets the latest version equal to #6.  The reference
     version does not change.  It is equal to #5.
```

(4) >>PURGE AFILE#R

Purges the current reference version, #5. This sets the reference version equal to the LATEST version which is #6 until the SETREF command is performed.

SETVERSION

Freezes changes to a file and increments the latest version number.

Syntax

```
SETV[ERSION] [filename]
```

Parameters

Filename	Name of the file to which you want to assign a version number. If the file name is not specified, Toolset will assign a version number to the file you are currently editing.

Discussion

The SETVERSION command freezes a version so that no further changes can be made and associates a new version number with the file. This command is valid only for files that are owned by the current workspace, and requires exclusive access to the file. Other users cannot access the file while the command is executing. New files are automatically asigned version 1. Further versions are assigned in ascending sequence starting with version 2. When a new version is assigned to a file, the contents of the previous version are frozen. You can have up to 32 active (non-purged) versions of a file at one time.

Each version logically includes active records from previous versions. For example, if version 1 has 10 records and 3 records are added to version 2, version 2 logically contains 13 records.

Example

```
Version 1:
1  This
2  is
3  version
4  1
5  of
6  the
7  file
8  YOURFILE

>>SETVERSION YOURFILE
```

Creates version 2 of YOURFILE. Version 1 above is now frozen and no further changes can be made to it. Only the LATEST version, version 2: version 2, can be modified after modifying line 4, deleting line 6, and adding lines 9 through 22, this is how version 2 would look.

```
1  This
2  is
3  version
4  2
5  of
7  file
8  YOURFILE
9  Notice
10 that
11 changes
12 can
13 be
14 made
15 to
16 it
17 because
18 it
19 is
20 the
21 LATEST
22 version

>>SETVERSION YOURFILE
```

Version 2 of YOURFILE has now been set. It contains records 1 through 5 and 7 through 22 since record #6 was deleted in version 2 above. You can only make changes in version 3.

SETVERSION FUNCTION KEY

Freezes changes to a file and increments the LATEST version number.

To use the SETVERSION function key:

(1) Position the cursor at any of the TSAM files in the Show Files menu and type a character in the box beside the file you choose.

(2) Press the Setv key

The current version of the file you specified is now frozen and no further changes can be made to that version.

To freeze the current version of file CRUN, enter a character in the brackets next to the filename and press the SETV key. The LATEST version will then be version #3.

SET WORKSPACE

Presents the Set Workspace menu and options.

Syntax

```
SET W[ORKSPACE]
```

```
HPTOOLSET                          Set
          Set OK                          Refresh          End Menu


SET WORKSPACE: ONE.MANU.TOOLS

* WORKSPACE USL FILE ..................  [ONEU.MANU.TOOLS                    ]

* WORKSPACE PROGRAM FILE ..............  [ONEP.MANU.TOOLS                    ]

* WORKSPACE LISTING DIRECTORY FILE .....  [ONEK.MANU.TOOLS                    ]
                          KEY FILE .....  [ONEK1    ]

* DICTIONARY ..........................  [DICT.GLOSS                         ]

* COMPILE PRIORITY ....................  [DS]

* PREP OPTIONS
[CAP=IA,BA,PH,MR;MAXDATA=30000                                               ]

* RUN OPTIONS
[LIB=G                                                                       ]
```

Figure 4-13. Set Workspace Menu

Example

Figure 4-13 shows an example of the set workspace menu. Here is a description of the fields in that menu.

Uslfilename — Allows you to modify the workspace USL file name. By default, Toolset takes the first seven letters of your workspace name and appends a U. This name cannot be given to any other files.

Progfilename — Allows you to change the workspace program file name. By default, Toolset takes the first seven letters of your workspace name and appends a P. This name cannot be given to any other files.

Listing Directory	The name of a KSAM file which maps between your edit, listing and program files. This file can be changed only when you are creating a new workspace. Any changes to it at a later time will be ignored.
Key File	The name of the key file associated with the Listing Directory file. This field can be changed only when you are creating a new workspace.
Dictionary	The name of the dictionary to be used with the GENERATE command. By default, this field will be blank and DICT.PUB will be used. If a dictionary name is entered, a file equation will be generated each time you enter your workspace which equates DICT.PUB to the given file name.
Compile Priority	The execution priority class for the compile (CS, DS, ES). The default is DS.
Prep Default Options	Allows you to make changes to the PREP command parameters. These parameters must be separated with semicolons with no semicolons preceding the first parameter. They take effect when prepping into the workspace USL and no other prep options have been specified.
Run Default Options	Allows you to make changes to the RUN command parameters. These parameters must be separated with semicolons. No semicolon should precede the first parameter. These options take effect when you are running the workspace program and no other run options have been specified.

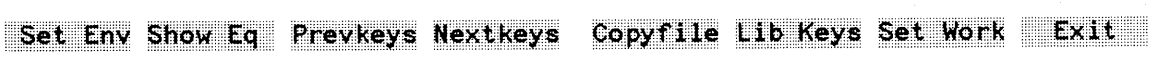
Discussion

The SET WORKSPACE menu is displayed when you create a new WORKSPACE command or function key. The workspace options which you can change are the workspace USL file name, the workspace program file name, the listing dictionary and key files, the dictionary name, the prep options and the run options. When you are satisfied with the menu, press the Set Ok function key. If you press the End Menu key, none of the changes will be saved and the new workspace will not be created.

SET WORKSPACE FUNCTION KEY

Pressing the Set Work function key displays the Set Workspace options menu which shows the current program translation options. This menu is initially displayed with default options. These options can be changed.

Main Utility Key Set



SHIFT

Allows you to shift a block of text to the right or left by a specified number of characters or from one column to another.

Syntax

```
SHI[FT] line rangelist [|FROM| column number] {|TO| column number}
                                               { L number         }
                                               { R number         }
```

Parameters

Line Rangelist	Identifies the range(s) of lines you want to shift. If more than one range is specified, they should be separated by commas. A line range has the form: line position [/line position] or ALL.
FROM Column Number	Identifies the column from which the specified range of lines will be shifted. The number specified must be greater than 0 and less than or equal to the record length of your file. If this parameter is omitted, the entire line will be shifted.
TO Column Number	The column number to which the specified range of lines will be shifted. The number specified must be greater than 0 and less than or equal to the record length of your edit file.
L Number	The number of columns you want the test shifted left. The number specified must be a positive integer less than the record length of your edit file.
R Number	The number of columns you want the text shifted right. The number specified must be a positive integer less than the record length of your edit file.

Discussion

Any text that is shifted past the first column or the defined LINELENGTH parameter (SET EDIT) will be truncated. Toolset automaticaly pads lines with blanks as necessary. This command can be cancelled with the UNDO command.

Example

(1) >>SHIFT 12/15 to 7

Shifts lines 12 through 25 so that they will begin at column 7.

(2) >>SHIFT 1/10 R 3

Shifts lines 1 through 10 to the right by 3 characters.

SHOW ACTIVITIES

Displays information about the current user environment.

Syntax

```
SHO[W] A[CTIVITIES]
```

Parameters

Activities — Lists whether Editing, Reading, Listing, Running and Compiling is taking place and the names of the files associated with those activities. It also lists the workspace that is open.

SHOW DEBUG

If there is a program that is currently being debugged, SHOW DEBUG will display the breakpoints and the names of any data items being traced as well as associated command lists.

Syntax

```
SHOW D[EBUG]
```

SHOW DEBUG FUNCTION KEY

Is the same as entering the SHOW DEBUG command.

Debug Edit Function Key Set

ShoDebug Listing Edit NextKeys Resume Refresh Help MainKeys

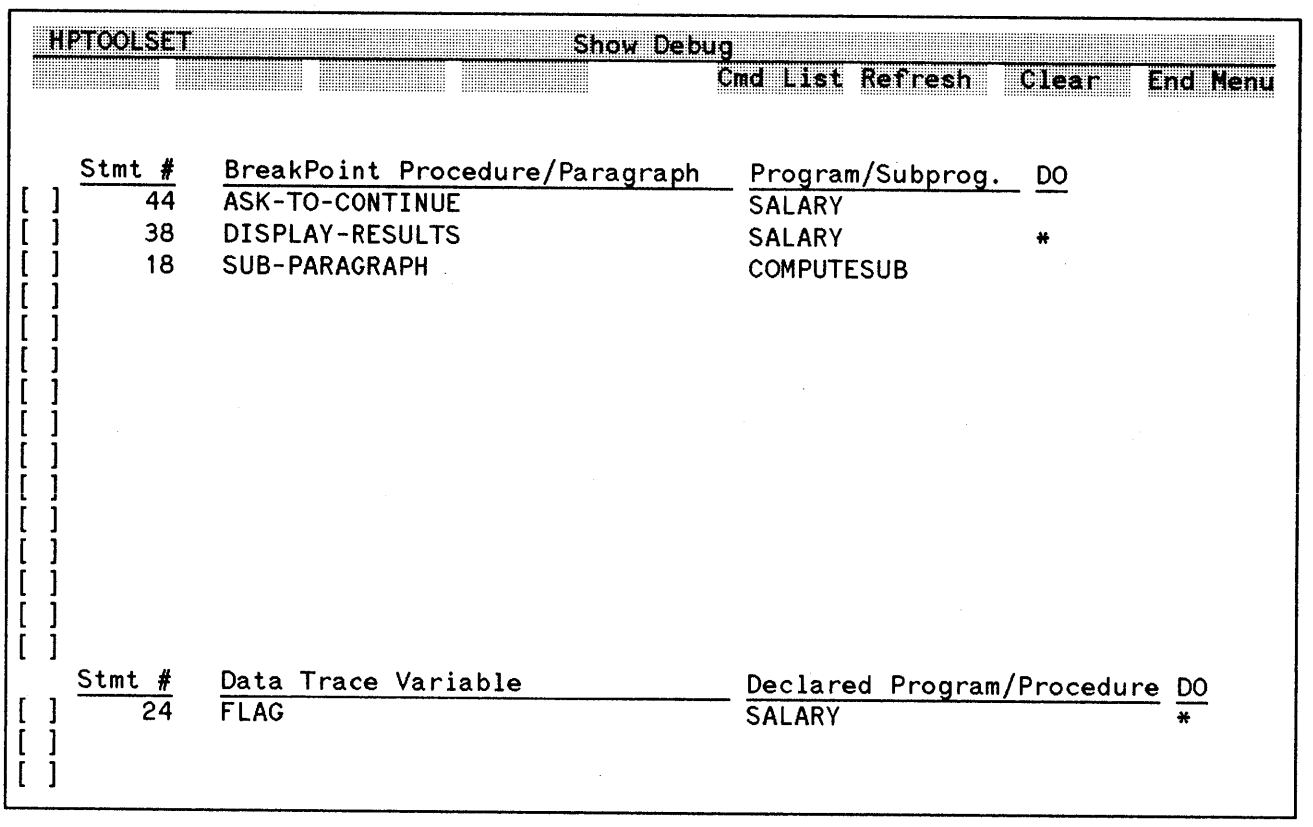


Figure 4-14. Show Debug Menu.

Breakpoints and datatrace variables can be cleared from this menu by entering a character in the box next to the breakpoint(s) to be cleared and pressing the CLEAR key. Toolset will erase the menu and display a message confirming the operation.

Displays all of the file equations set up by the USE commands in effect for the current workspace.

Syntax

```
SHOW E[QUATES]
```

Discussion

There is one file equation for each USE comand issued that includes a formal designator.

Example

```
>>USE FRAME FOR XYZ

>>SHOW EQUATES
```

```
HPTOOLSET                    Show Equates


     "USE" File Equations for Workspace: ONE.MANU.TOOLS

  :FILE APPLES    =ORANGES.PADULA.TOOLS#3
        FILEX     =FILEX.MANU.TOOLS#4
```

Figure 4-15. Show Equates Display

SHOW EQUATES FUNCTION KEY

Is the same as entering the SHOW EQUATES command.

SHOW EQUATES

This key is available in the Main Utility Set. It displays all of the file equations that have been set up by the USE commands and are in effect for the current workspace.

Main Utility Function Key Set

SHOW FILES

Displays a list of all owned and used files for a particular workspace.

Syntax

```
SHO[W] F[ILES] [workspacename]
```

Parameters

Workspacename	Identifies the workspace for which the owned and used files are displayed. If no workspace is specified, the display will be for the current workspace.

Discussion

If no workspace name is given, it displays the contents of the current workspace. It also displays the:

(1) name of the owner workspace for all used files

(2) number of active versions of each owned file

(3) REFERENCE version number of each owned file

(4) LATEST version number of each owned file

(5) type of each file (USE, OWN, USL or PROG)

SHOW FILES FUNCTION KEY

Is the same as entering the SHOW FILES command without the workspace parameter.

Main Edit Function Key Set

The Show Files function key is available in the Main Edit Function Key Set.

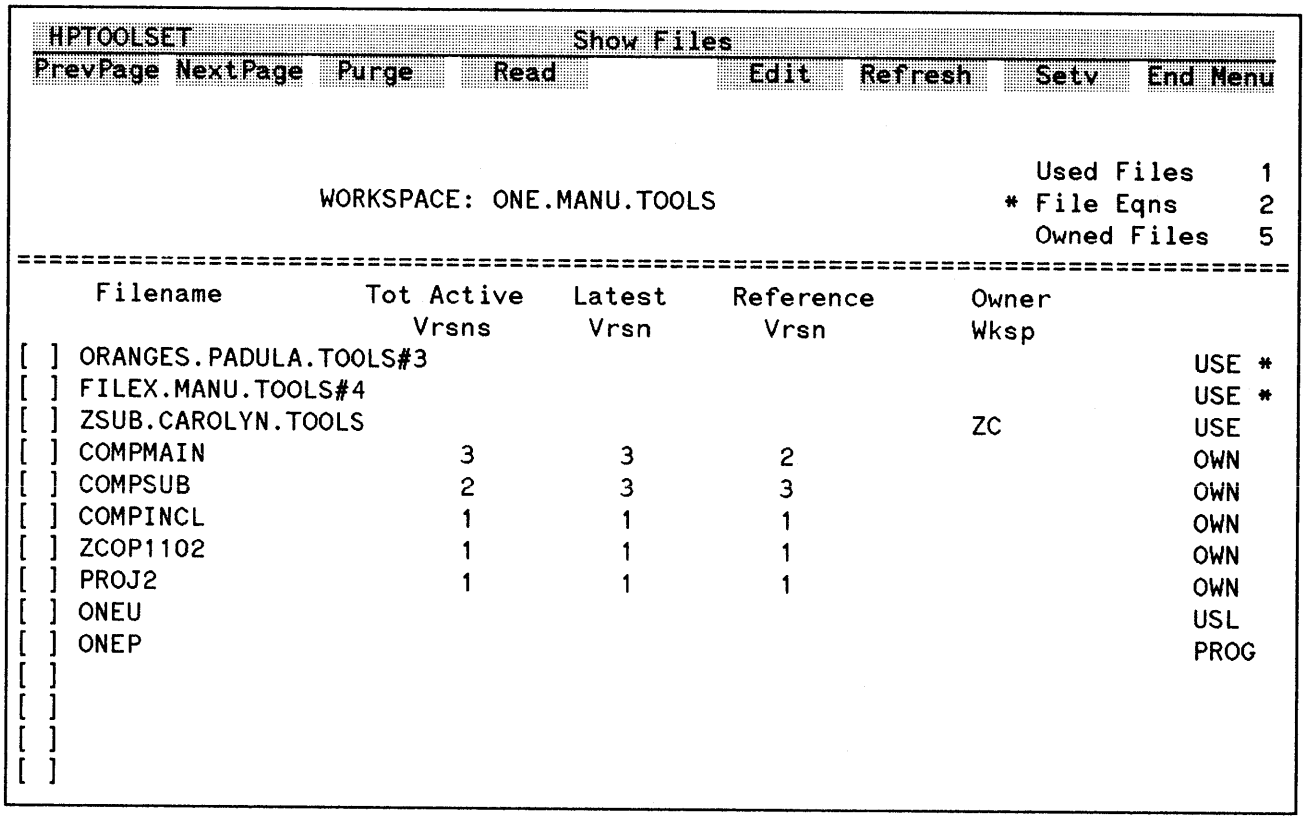


Figure 4-16. Show Files Menu

The Show Files Menu displays the following:

(1) Workspace	Is the name of the workspace for which the files are being listed.
(2) Name of Owner Workspace	Is the name of the workspace that ownes the used files.
(3) Number of Active Versions	Is the number of versions not purged from the file.
(4) LATEST version	Is the version number assigned with the last SETVERSION command.
(5) REFERENCE version	Is the version which users other than the file owner use by default.
(6) OWN, USE, USL, PROG	Is the type of file. The USL and Program file names are defined when your workspace is created. The default is to append a U or a P to the first seven characters of the workspacename.
(7) *	Indicates that a file equation is associated with the USE file.

Pressing the Show Files function key or entering the SHOW FILES command produces a secondary function key set containing the Edit, Read, SetVersion and Purge operations. Any of these operations can be performed by typing a character in the box beside any file or pressing the appropriate function key.

SHOW LABEL

Displays any comments associated with a version of a file.

Syntax

```
SHO[W] LA[BEL] filename [version range]
```

Parameters

Filename	Name of the file for which comments will be shown.
Version Range	Takes the form: #n or #@ where: n = integer, L[ATEST] or R[eference]. The @ - all active versions.

Discussion

If no version is specified for filename, the label for the LATEST version will be displayed if you own the file. The label for the Reference version will be displayed if you do not own the file or if a USE has been issued.

Example

```
>>SHOW LABEL XFILE#3/#R

-->Version #3

Version 3.  3/22/83

-->Version #4

Version 4.  6/12/83  -  Prereleased version

-->Version #5

Version 5. 7/20/83

-->Version #6

version 6. 10/10/83
>>
```

Displays the copylib name, key file name and the names of all the modules in the copylib.

Syntax

```
SHO[W] LI[BRARY] [libname]
```

Parameters

Libname	Is the name of the library. If you omit it, Toolset will look for a workspace default library. If no default is found, Toolset will ask for the library name.

Example

```
>>SHOW LIB COPYLIB

-->LIBRARY FILE: COPLIB.PUB.PROJECT

   KEY FILE: COMPKEY.PUB.PROJECT

-->LIBRARY MODULES:

   INTCOMP

   REALCOMP
      .
      .
   ANYCOMP

-->REALCOMP is open for editing
```

This last line will be displayed only if a module within the library was opened for editing when the command was entered.

SHOW LIB FUNCTION KEY

Is the same as the SHOW LIB command without parameters.

SHOW SOURCE

Displays workspace source files.

Syntax

```
SHO[W] S[OURCE]
```

SHOW SOURCE FUNCTION KEY

Displays a list of your workspace source files and information about each of them.

Main Program Function Key Set

ShoSourc | Listing | PrevKeys | NextKeys | Compile | DebugKey | | Exit

```
HPTOOLSET                          Show Source
PrevPage NextPage Compile Listing          Edit  Refresh   Go   End Menu

   SOURCE FILES for Workspace ONE.MANU.TOOLS

    PROGRAM ID/PROCEDURE          LANG    TYPE     SOURCE FILE      LISTFILE

[ ] SALARY                        COBOL   MAIN     COMPMAIN         ONE61031
[ ] COMPUTESUB                    COBOL   SUBP     COMPSUB          ONE61086
[ ] COMPUTESUB                    COBOL   INCLUDE  COMPINCL
[ ]                               COBOL   LIB MOD  ZCOP1102
[ ]                               PASCAL  MAIN     PROJ2
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
[ ]
```

Figure 4-17. Show Source Display

The Show Source Display gives the following information for each file belonging to your current workspace:

(1) Program-ID/Procedure	This is the COBOL program-id or Pascal procedure name for the given source file. It will appear only after the file has been compiled.
(2) Lang	Identifies your programming language as COBOL, Pascal or other.
(3) Type	Identifies your program as main, subprogram, include, lib module or other.
(4) Source File	Is the name of your source file.
(5) Listfile	Is the name of the compiler listing file for your source program. This will appear only after the file has been compiled.

To perform any of the program translation operations available in the function key display, type a character in the box beside the source file and press the appropriate function key.

STORE

Stores all owned files, the USL file, program file, listing files and directory information for the specified or currently active workspace.

Syntax

```
ST[ORE] [workspacename]
```

Parameters

Workspacename	Is the name of the workspace to be stored on tape. If no name is specified, the currently active workspace will be stored.

Discussion

If you are storing the current workspace, all ongoing activities will be terminated before the STORE is performed.

Example

```
>>STORE WRKSPCX

Stores all files associated with WRKSPCX onto tape.
```

Accesses the MPE Debug Utility.

Syntax

```
SYS[DEBUG]
```

Discussion

The SYSDEBUG command allows you to access MPE DEBUG from Toolset. Exiting MPE DEBUG does not automatically return control to Toolset. Your program resumes execution and control is returned to Symbolic Debug only when a Toolset interrupt is encountered or when execution of your program is terminated.

MPE DEBUG cannot be accessed from Toolset when your program is suspended at an active breakpoint, until the breakpoint is cleared.

Example

```
>>SYSDEBUG
***Breakpoint conflict at this location.(744)
-->Clear current Toolset breakpoint?YES
-->Breakpoint cleared.
*BREAK* 0.2763
?
```

TRACE

Identifies each subprogram, section, paragraph or procedure before it executes.

Syntax

```
T[RACE] [OFF]
        [ON ]
```

Parameters

OFF	Disables the TRACE facility.
ON	Turns on the TRACE facility. This is the default.

Discussion

For COBOL programs, this command displays the name of each subprogram, section, and paragraph when encountered in your program at run time. For Pascal programs, the TRACE command causes an identifying message to be displayed at the start and end of each procedure.

TRACE FUNCTION KEY

The Trace Function Key is available in the Debug Utility function key set. It functions the same as the TRACE command for both COBOL and Pascal programs.

Allows you to cancel the last CHANGE, DELETE, MOVE, SHIFT or MODIFY command entered.

Syntax

```
UN[DO]
```

UNDO FUNCTION KEY

The Undo function key is available while you are editing in visual mode. It will Undo the modifications which were caused by pressing Return or a function key.

When the Undo key is pressed, your screen is repainted. The Undo buffer will be reset when another edit modification key is pressed or a new screen is transmitted.

Example

```
Example 1

>>Modify 133
133     procedure_name : packed array[1..50] of ' '..'Z'
                                                 R3
133     procedure_name : packed array[1..30] of ' '..'Z'
>>UNDO
133     procedure_name : packed array[1..50] of ' '..'Z'
>>
```

```
Example 2

>>DELETE 200/210
200
201     var
202       mpefile :
203        record
204          data     :  packed array[1..72] of char;
205          line_no  :  packed array[1..8] of '+'..'9'
206       end;
207
208     const
209        mpefile_const  =  'mpefile ';
210
>>UNDO
210
209        mpefile_const  =  'mpefile ';
208     const
207
206         end;
205           line_no  : packed array[1..8] of '+'..'9'
204           data     : packed array[1..72] of char;
203         record
202        mpefile :
201     var
200
```

The Undo key is located in two function key sets:

Edit Move Set

Mark	Clr Mark	PrevKeys	NextKeys	Move	Copy	Undo	End Edit

Edit Print Set

Set Edit	Print	PrevKeys	NextKeys	Cmd Mode	Refresh	Undo	End Edit

Allows two workspaces to share a file and specifies the version to be shared.

Syntax

```
USE filename1[version designator] [,filename2[version designator]]...

USE [formal1 |FOR|] filename1[version designator]
    [,[formal2 |FOR|] filename2[version designator]] ...
```

The first format allows you to share a file that is owned by another workspace. A specific version of the file can be shared if you specify a version number. If you do not, a default access version will be assigned. The default version of a shared file is the Reference version.

The second version is an extension of the first and allows you to specify a formal designator. It generates a file equation between formal and filename so you can refer to the shared file by a different name.

Parameters

Filename1	Name assigned to the file you want to share.
Version Designator	Takes the form: #n where n is an integer, LATEST(L) or REFERENCE(R). Allows you to specify which version of a file you want to share. If a version designator is not specified, Toolset will use the default (Reference) version.
Formal	Is the formal file designator. It may be up to 8 alphanumeric characters long.
Filename	Actual file designator for the file equation.

Discussion

Toolset automatically generates the file equations implied by the second format every time you enter the associated workspace. The USE command need only be issued once per file.

A file version that is specified in the USE command can be overridden by specifying another version number in a subsequent Toolset operation. For example, if you have entered the USE command:

>>USE ZFILE#2

Version 2 of ZFILE will be accessed by default. You can override this default by specifying a new version.

>>CONVERT ZFILE#4 TO NEWFILE. Converts version #4 of ZFILE overriding the version specified in the USE command.

Example

```
(1) >>USE JFILE FOR KFILE.group
          .          .
          .          .
        Work-        .
        space A      .
                     .
                   Work-
                   space B
```

Allows Workspace A to share KFILE owned by Workspace B. Because Workspace B resides in a different group, KFILE must be qualified with its group. Since no version is specified for KFILE, Workspace A uses the default version (Reference version).

Toolset sets up a file equation for JFILE and KFILE.group.

```
(2)  >>USE GFILE#5
```

Indicates that version 5 of GFILE is to be used by default when GFILE is opened. This command can be entered by the owner of GFILE or a user sharing GFILE.

WORKSPACE

Allows you to create or change workspaces.

Syntax

```
W[ORKSPACE] [workspacename]
```

Parameters

Workspacename	A filename representing the name of the directory within which your program is being developed. This name needs to be fully qualified if it does not reside in your logon group and account.

Discussion

When you change workspaces, the one that you are in will be closed along with all o[pen files. All file equations generated through the USE command will be reset and the information in the Workspace Directory will be updated.

The Workspace function key operates like the WORKSPACE command without parameters. You will be prompted for the workspace name. If the workspace you specify does not exist, a new one will be created. The Set Workspace menu will be brought up to allow you to define the default options.

Workspace Function Key Set

The Workspace function key is in the Workspace Function Key Set.

XEQ

Allows input from a specified file.

Syntax

```
X[EQ] filename[version designator]
```

Parameters

Filename	Either an ASCII or TSAM formatted file from which Toolset accepts commands.
Version designator	Particular version of the input file. Has the form #n or #LATEST or #REFERENCE.

Discussion

The XEQ command directs Toolset to accept command input from a TSAM or ASCII file. The commands will be echoed on your screen as they are executed, unless a SET ECHO=OFF has been done. If an error or a command that displays a menu is encountered, the remainder of the XEQ file will not be executed.

Example

```
>>XEQ CMDFILE#2

SET QUIET=ON

SET PAGESIZE=40

WORKSPACE COMPUTEW

>>
```

WORKSPACE & FILE MANAGEMENT

SECTION 5

WORKSPACE

Creating & Accessing a Workspace

For a description of how to create and access a Workspace, see Section 1 of this Reference Manual.

Displaying the Contents of a Workspace

There are five main commands which allow you to display the contents of a Workspace:

1) Show Files	Gives a list of all Use files, Owned files, the USL file and the program file.
2) Show Source	Gives the language, type and name of the associated listing file for each owned file.
3) Show Equates	Displays all file equations that were generated as a result of the USE command.
4) Show Library	Gives the name of the default copy library and module names within it.
5) Set Workspace	Gives the name of the USL file, program file, listing directory file, key file, dictionary and compile priority. It also gives the default prep and run options and allows you to change them.

For details regarding each of these commands, see Sections 3 and 4 of this Reference Manual.

Adding Files to a Workspace

There are two types of source files within a Workspace - Owned files and use files. Owned files belong to the current Workspace and can be edited within it. Use files are outside of the Workspace, e.g., MPE files or files created in a different Workspace.

The following commands allow you to add new owned files to your workspace:

EDIT Command	When you enter the EDIT command with a new name, a new file will be created with that name.
COPYFILE Command	Is used to make a copy of an existing TSAM file.
CONVERT Command	Is used to convert an existing MPE or KSAM file to a TSAM file.

Use files are added to the Workspace by entering the USE command.

Deleting Files from a Workspace

Owned files are deleted when you execute the PURGE command. You should do this while in the workspace so the owned file entry can be deleted from the Workspace Directory.

To delete a USE file, you must execute the DISCARD command.

TSAM FILES & VERSION MANAGEMENT

Whether you operate in a single or multiple programming environment, programs undergo multiple changes. TOOLSET recognizes and manages each changed file as a version. As you make changes to your file (additions, deletions and updates) ,these changes are kept as versions of the same file.

Adding Versions

The Toolset SETVERSION command freezes changes to a file and increments the LATEST version.

Versions are referred to by number and the most recent copy of a file is known as the LATEST version. When a file is created, it is version 1. The first SETVERSION command adds version 2 and subsequent SETVERSION commands add versions 3, 4 and so on. Each version logically includes records from previous versions.

You can list all changes with the LIST CHANGE command.

Latest Version

The Latest version of a file is the working or most recent version of a text file, and is the only one that can be modified. Modifications can only be made by the owner of the workspace.

Each text file can have up to 32 active versions. An active version is one that has not been purged.

Reference Version

The Reference version is the version that is accessed by users other than the owner of the workspace. It must be defined with the SETREF command. If you do not define the Reference version the LATEST version will be the default.

Version Access

Versions are accessed according to the access pointer setting. The owner of a file that is shared can set the access pointer with the SETREF command. This will specify the version of the file that is to be referenced by other users. The version that is set with the SETREF command is the Reference version. If no version has been set by the file owner, other users access the LATEST version by default. Only the LATEST version of a file can be modified.

If you own a file, you always access the LATEST version of the file unless you issue a USE or request a particular version. While the LATEST version is being edited, it is locked and no other users can access it. To enable other users, including the compiler, to read the LATEST version, you must end the edit operation with the END EDIT command or function key. By default, you compile the LATEST versions of the files you own and the most stable (Reference) versions of the files you do not own or have issued a USE on.

A USE overrides an OWN. If you issue a USE command for a file which you own, you access the Reference version of the file by default until a DISCARD is issued to negate the USE.

If an abnormal termination occurs while you are accessing the LATEST version of the file, the file remains locked. It must be recovered before it can be accessed.

A LIST CHANGE command is available to provide a listing of the changes from one version of a file to a higher version.

EDITING

SECTION 6

OPENING AN EDIT FILE

Before you can edit a file, you must either open an existing one or create a new one. To open an existing file for editing, type the EDIT command following the Toolset prompt >> or press the Edit function key. If you do not specify the filename, you will be prompted for it. If you are not calling up an existing file, you will be asked if you want to create a new one. For new files, a SET options menu is displayed in which the file's dimensions, such as LINELENGTH, can be defined. When you have specified all the options you want, press Set OK.

When you are creating a new COBOL main or subprogram, the first four lines will be created for you and will appear on your screen. For new files which are written in languages other than COBOL, the screen will be blank and numbers will not be displayed until your screen is repainted.

To edit a file, the file must be in TSAM (Toolset Access Method) format and exist in the current Workspace. Only one file can be open for editing at a time.

Since edit files are owned by the current Workspace, all edit files within a Workspace belong to the same group and account as the Workspace. The Workspace group and account may be different from your group and account.

EDITING MODES

There are two editing modes: Visual Mode and Command Mode.

Visual Mode

Visual Mode is used to create, modify, and read any TSAM files. It is screen oriented. Editing functions are performed by positioning the cursor and typing over the text which is displayed on the screen.

Visual Editor Function Keys

There are five function key sets available in the visual editor. They are displayed in visual mode only.

Edit Find Set

Edit Browse Set

PrevPage	NextPage	PrevKeys	NextKeys	Find-N	Find+N	Set N	End Edit

Edit Move Set

Mark	clr Mark	PrevKeys	NextKeys	Move	Copy	Undo	End Edit

Edit Add Set

Mark	Clr Mark	PrevKeys	NextKeys	Add	Generate		End Edit

Edit Print Set

Set-Edit	Print	PrevKeys	NextKeys	Cmd Mode	Refresh	Undo	End Edit

Terminal Keys for Editing in Visual Mode

Key	Function
Home Cursor	Moves the cursor to the beginning of the display memory of the terminal.
Up Arrow	Moves the cursor up a row. If the cursor is at the bottom of the screen, it will wrap around to the top of the screen.
Down Arrow	Moves the cursor down a row. If the cursor is at the bottom of the screen, it will wrap around to the top of the screen.
Left Arrrow	Moves the cursor one column to the left. If the cursor is at the left margin, it will wrap around to the right.
Right Arrow	Moves the cursor one column to the right. If the cursor is at the right margin, it wraps around to the left.
Roll Up	Moves the display memory up a row. The cursor remains stationary.
Roll Down	Rolls the display memory down a row. The cursor remains stationary.
Clear Display	Deletes all lines of text displayed beyond the cursor.
Insert Line	Inserts the line containing the cursor. All lines following it are rolled down one line. A blank line is inserted just above the line containing the cursor and the cursor moves to the leftmost column. Line numbers are not displayed unless you press the Refresh key.
Insert Char	Inserts new characters in the line at the current cursor position. The characters to the right of the cursor are shifted right as new characters are added, and are truncated if the line overflows. To end the Insert Character function, press the INSERT CHAR key again.

Del Line	Deletes the line at which the cursor is positioned and the lines below are rolled up. The cursor is repositioned in the first column.
Del Char	Deletes the character at the current cursor position. The text to the right of the cursor is shifted left as the characters are deleted.
Tab	The cursor is moved forward to the next tab (using TAB) or backwards to the previous tab (using CNTL-TAB). Tabs should only be used in Visual Mode. Tabs are set and cleared by using the SET EDIT options menu. The CLEAR TAB and SET TAB keys should not be used.
Backspace	The cursor is moved one column to the left. If the cursor is in the first column, it does not move.

The visual editor has a maximum limit of 60 lines per screen. If that limit is reached while you are inserting lines, your screen is repainted. By default, 20 lines of text will be displayed. To override this default, set PAGESIZE = any integer between 1 and 40 inclusive. If you have more than 20 lines displayed, use the terminal ROLL UP, ROLL DOWN, NEXTPAGE and PREVPAGE Keys to scan the text.

To display a new page of text, use the function keys labelled NextPage, PrevPage, Find+N, and Find-N. A Find key is also available to locate specific strings or lines you want to see within the file.

Some of the edit function keys require that you specify one or more lines or character positions before using them. Those keys include MOVE, COPY, GENERATE, and ADD. Lines are designated by positioning the cursor and pressing the Mark function key twice. Character positions are designated by positioning the cursor at the desired character and pressing the Mark function key once.

Command Mode

As you can see from our explanation of the visual editor function keys and terminal keys for editing in visual mode, these are the most efficient means for editing. You can, however, edit your source files by entering commands in Command Mode.

You can access Command Mode from Visual Mode through the use of the command mode function key.

TEXT MODIFICATION COMMANDS.

The following are the Text Modification Commands available while a file is being edited:

Copy	Find
Move	Undo
Shift	Renumber
Change	Modify
Delete	Generate
List	Add

The Use of the Add Command

The ADD command is available for editing a file while you are in Command Mode. However, Toolset does not show you the Command Mode double prompt. Instead, you are prompted with line numbers.

Editing with Other Languages

Toolset can be used with languages other than COBOL and Pascal. If you are programming in another language, the same function key sets and menus are available. The SET options for files in other languages are, however, different from those for COBOL and Pascal.

COPYLIB EDITING

Function Description

The Copylib Editing Function is a substitution for Cobedit. It provides Toolset users with all the capabilities that are currently provided by Cobedit for creating, editing, and managing Copylibs. It will allow you to:

- Create a new Copylib
- Enter a new module into a Copylib
- Copy a module from another edit file
- Edit a module
- List the contents of each module in a Copylib
- Display the file attributes
- Delete modules from the copylib

Copylib User Interface

Six new comands are available in Toolset to support Copylib Editing. These commands may be entered while in Command Mode or by pressing the appropriate Copylib function key.

The Copylib function keys are accessed by pressing Lib Keys in the Main Utility Set.

COPYLIB COMMANDS	CORRESPONDING COPYLIB COMMAND FUNCTIONS
Set Lib	Defining a default library and building a new library
Libedit	Editing a Copylib Module
Lib List	Listing one or all of the Copylib Modules
Show Lib	Displaying Copylib attributes and module names
Libpurge	Purging a Copylib Module
Libcopy	Copying a module into a Copylib

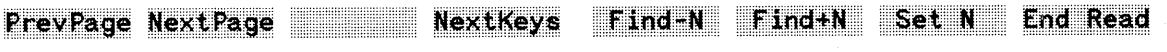
Reading a Source File

You can use the READ command to display the contents of a source file in either Visual or Command Mode. One file can be open for reading while another is open of editing. Any versions of the file can be open for reading. No modifications can be done to files which are open for reading only. In Command Mode only the FIND and LIST commands are available.

The following function keys are available when you are reading a file in Visual Mode:

Read Set 1

Read Set 2



PROGRAM TRANSLATION

SECTION 7

PROGRAM TRANSLATION

This section contains information on preparing and compiling.

HOW TO COMPILE FILES

There are three ways to compile source files. You can use the: Compile function key, COMPILE command or the Go function key. The COMPILE command and function key accomplish the same objective. When executed, they activate a process that manages the compile process. All of this occurs in background mode and is not apparent from looking at your terminal screen. Therefore, you are free to perform other functions while this process is taking place.

When you use the Compile command, you need not specify the USL file into which you want the source files placed. By default, source files which are owned or used by the current Workspace will be put into the Workspace USL file. Source files that are not so owned or used will be put into $OLDPASS by default.

The Go function key performs three functions. It allows you to prep, compile and run your program by pressing a single function key.

Every 90 secords or everytime you press return, Toolset will check to see if your compile has completed. When it is finished, a message will be displayed on your screen, giving you a summary of compile errors and warnings.

Auxiliary KSAM Files

When a Workspace is created, Toolset creates an auxiliary KSAM file that is associated with the Workspace. This is the Listing Directory file named in the SET WORKSPACE menu. This file is used by Toolset to associate the source files with the listing and program files.

The Compiler Listing

As the source file is compiled, a listing is created through a message file. If the source file is owned by the current Workspace, this listing will be put into a TSAM file.

The Compiler listing can be displayed through the use of the LISTING command. If you want a hard copy of this listing, you should type PRINT ON before doing the compile or press the Print Key while the listing is displayed on your screen. If the source file was not owned by the current Workspace, only the hardcopy listing can be generated. The hard copy listings will be sent to the line printer unless a file equation has been set up for the formal designator TOOLLIST.

The LISTING command or key is used to view the TSAM listing files that were created during the compile. The listing can be displayed in Visual or Command Mode. When displaying the listing in Visual Mode, function keys are available which allow you to locate and correct compile errors easily. These keys are the FIND ERR and the EDIT keys.

Since COBOLII groups the error messages together at the end of the listing, the Find Err key will position you at the end of the file. If you want to view a line which contains an error, you should mark the error by positioning the cursor at the error line, and pressing the Mark key twice. Then press the Listing key. The compile listing would be displayed on your terminal screen at the line that caused the error. Similarly, if you wish to edit a line that contains an error, position the cursor at the error, mark it by pressing the MARK key twice and then press the Edit key. Your edit file will be displayed at the point in your file where the error or warning occurred.

Pascal has the error messages merged into the listing where they occurred. The FIND ERR key will sequentially locate compile errors and position you five lines before the error message. If you wish to correct the mistake in your source file, simply press the Edit key. Your file will be displayed at the point where the error occurred. After correcting the error in your edit file, you can return to the listing by pressing the Listing key.

HOW TO PREPARE FILES

There are three ways to prepare USL files. You may use the PREP command, Prep function key or Go function key. The PREP command and function key prepare a USL file and create a program file.

Ending Compiles

If you want to end a source compile before it has completed, type END COMPILE following the >> prompt on your screen. The compile will end and you will be returned to Toolset. You can also type END without parameters and be prompted for which of the ongoing TOOLSET activities, including compiles, you wish to end.

Toolset does not allow you to EXIT while you are compiling. If you enter the EXIT command during a compile, you will be asked if you want to terminate all compiles. A 'yes' response will end your compiles, terminate Toolset and return you to MPE. If you answer 'no', your compile will continue and you will remain in Toolset.

NOTE

If you terminate your compile with either the END COMPILE or EXIT command, you may need to recover your source file. The RECOVER command in Section 4 of this Reference Manual will provide the necessary information.

SYMBOLIC DEBUG

SECTION 8

Symbolic Debug is one of the programming tools that make up the Toolset utility. It allows you to interactively monitor your program's execution and debug your program without having to know memory locations or convert source statements into octal code offsets. Instead, breakpoints can be set and execution flow traced by referencing the section and paragraph names from the Procedure Division of COBOL programs, the procedure name from Pascal programs or the compiler generated line numbers from programs written in either language. Data items can be monitored, displayed and changed by using their names.

The Symbolic Debug feature works in conjunction with other Toolset features. Toolset commands can be entered during the debugging phase of program development.

HOW TO RUN SYMBOLIC DEBUG

To run your program with the Symbolic Debug feature, you must first compile your source files with a special control option. For COBOL programs, you must enter $CONTROL SYMDEBUG before the first executable statement in your source files. For Pascal programs, you must enter $SYMDEBUG$ before the declaration statements in the source file. In both cases, the Symbolic Debug option is then automatically activated when your program is compiled and prepared.

```
HPTOOLSET                                    Edit Find
 Find   Change  PrevKeys NextKeys          Cmd Mode Listing   Help   End Edit


 1     $CONTROL USLINIT, SYMDEBUG
 2      IDENTIFICATION DIVISION.
 3      PROGRAM-ID.
 4          SALARY.
 5      AUTHOR.
 6          JANE PROGRAMMER.
 7      DATE-WRITTEN.
 8          5-27-81.
 9
10      ENVIRONMENT DIVISION.
11      CONFIGURATION SECTION.
12      SOURCE-COMPUTER.
13          HP-3000-SERIES-64.
14      OBJECT-COMPUTER.
15          HP-3000-SERIES-64.
16
```

Figure 8-1. Indicating SYMDEBUG in Control Statement (COBOL)

```
HPTOOLSET                                    Edit Find
 Find   Change  PrevKeys NextKeys          Cmd Mode Listing   Help   End Edit


 1      $USLINIT$
 2      $SYMDEBUG$
 3      $TITLE 'Globals'$
 4      $PAGE$
 5      PROGRAM project1 (INPUT, OUTPUT);
 6
 7      CONST  page_size = 256;
 8             last_frame = 7;
 9             last_segment = 31;
10             last_page = 3;

                    .
                    .
                    .
```

Figure 8-2. Indicating SYMDEBUG in Control Statement (Pascal)

Breakpoints

If your program includes the $CONTROL SYMDEBUG (COBOL) or the $SYMDEBUG$ (PASCAL) as a control statement, it automatically stops at the first executable statement in the Procedure Division (COBOL) or the main program (Pascal) when it is run inside Toolset. At this point, you can execute other sysmbolic debug commands (including setting additional breakpoints) or enter other Toolset commands.

SYMBOLIC DEBUG COMMANDS

The Symbolic Debug commands are:

- AT
- BREAK
- CALLS
- CLEAR
- DATATRACE
- DISPLAY
- END RUN
- MOVE
- RESUME
- RETRACE
- SHOW DEBUG
- SYSDEBUG
- TRACE

For an explanation of these commands, see their alphabetical locations in Section 4 of this Reference Manual.

SYMBOLIC DEBUG PARAMETERS

LIST OF TERMS

(1)

Data-Item Type	Selector Symbol (Pascal)	Example
ARRAY	[,]	ARRY1 [1,2,3] ARRY1 [1][2,3]
RECORD	.	MyVar.Field4

Toolset also allows the following selectors even though they are not permissible in standard Pascal:

```
ARRAY elements may be selected as (,):

A(1,3)

RECORD elements may include OF or IN in the syntax:

DISPLAY F2 IN F1 IN VAR1
DISPLAY F2 OF F1 OF VAR1
```

(2) Figurative Constant (Pascal)

Name	Type	Value
FALSE	BOOLEAN	0
TRUE	BOOLEAN	1
MININT	INTEGER	-2147483648
MAXINT	INTEGER	2147483647
NIL	POINTER	32767

(3) Literals

Non numeric literals must be delimited by single quotes (') or double ("). Numeric literals which do not require quotes, may be preceded by plus (+) or minus (-):

Non-numeric	"Hi There"
	'So Long'
Numeric	25
	+13
	-100

(4) Stmt-no (Pascal)

Stmt-no is the compiler generated integer which appears on the listing in front of each statement in a Pascal or COBOL program. When specifying a statement number in Symbolic Debug, enter that number (up to 5 digits) preceded by a pround (#) sign. Statement numbers may only have one qualifier.

Statement numbers (stmt-no) may be qualified by appending them to a program-id, level-1 procedure name or any level-n procdure name inside the current procedure name. reference the currently executing program. If a statement number is in the level-1 procedure and is not qualified, it references the corresponding stmt-no of that procedure.

(5) Location (Pascal)

A location is any program, procedure or Pascal label name. Locations can be referenced with a statement number:

ProcName + Stmt-No

ProcName - Stmt-No

LabelName + Stmt-No

LabelName - Stmt-No

If the specification is of the form x.y.z, only z can be a label name and x and y must always be procedure names.

(6) Procname

A procedure name in the source program. Procedure names are qualified in the following ways:

```
ProcName-LevelB OF Procname-LevelA
                IN

ProcName-LevelA.ProcName-LevelB

ProcName-LevelA...ProcName-LevelB
```

where ".." indicates that LevelB is one or more levels deep inside LevelA.

To avoid ambiguities, fully qualify procedure names.

SYMBOLIC DEBUG FUNCTION KEY LOOP

A two set function key loop is available whenever you run a program with Symbolic Debug:

Debug Edit Set

ShoDebug	Listing	Edit	NextKeys	Resume	Refresh	Help	MainKeys

Debug Utility Set

Trace	Retrace	Calls	NextKeys	Resume	At Next	Clear	End Run

When in the Debug function key loop, you can return to the Main function key loop by pressing f8 in the Debug Edit Set. You can then re-enter the Debug loop by pressing the Debug key in Main Program Set 2 (Displayed in Symbolic Debug only).

When your program completes execution or is terminated by the **END RUN** command, you will be taken from the Debug loop to the Main function key loop.

To see all the Toolset function key sets, turn to Section 2 of this Reference Manual.

Control Y

Entering control y when programming in COBOL returns control to Toolset, unless you code control y to perform a different function. In that case, the coded function takes priority.

For Pascal programs, control y has no effect if you are in a construct, such as a WHILE, that does not call a procedure or function of your program. Once such a call is encountered, control y returns control to Toolset.

Process Handling

Symbolic Debug handles up to 3 processes simultaneously when process handling is employed. If 3 processes are using Symbolic Debug, any other processes created are run without Symbolic Debug, regardless of how they are compiled and prepped. The user is responsible for the timing and the interaction of his program.

Xtraps

Toolset arms XCODETRAP, XLIBTRAP and XSYSTRAP for user programs. If your program terminates abnormally because of the code exception or run time error detected in the compiler software library or system intrinsic calls, control is returned to Symbolic Debug. An abort error message is displayed at this time along with the currently executing calls. The DISPLAY, RETRACE, LISTING or CALLS command can be entered to find the reason for a program abort. Because the state of the program is unpredictable, a RESUME command is not allowed at this point.

If you choose to arm XCODETRAP, XLIBTRAP or XSYSTRAP with your own trap handlers to perform a different function, that function has priority.

Debugging Files Outside Workspace

If you want to run Symbolic Debug on a program file other than the one in your current workspace, you can use one of two options:

(1) Enter the workspace of that program file or issue a USE command for the program file so that it is known to your workspace.

(2) Issue a :RUN progfilename

Debugging Files Outside Toolset

The Toolset Symbolic Debug utility can be used for programs you have created and modified with an editor other than the Toolset editor. To run Symbolic Debug on these programs, use the following steps.

(1) Use a $CONTROL SYMDEBUG statement in your COBOL source file or $SYMDEBUG$ statement in your Pascal source file.

(2) Compile your source file outside of Toolset. Use the MPE compile command.

(3) Prep your USL file outside of Toolset using the MPE PREP command.

(4) Enter Toolset and issue a Workspace command, giving the name of a "dummy" Workkkspace.

(5) Do a USE command on the program file name used when you prepped the USL file.

(6) Run your prograsm using the program file name :Run programe

No list file is generated when source files are executed in this way, so you will not be able to perform debug operations by referencing line numbers or by Marking. You can perform debug operations through symbolic referencing with Debug commands.

>>AT PARA-TWO

NOTE

If you are running a program with symbolic debug that has a COBOL outer block with Pascal procedures, you cannot DISPLAY, MOVE or DATATRACE any Pascal procedure, parameter or variable that references a Pascal global type. Further, if you do a Pascal subprogram compile that references a global type that has been declared in the subprogram's outer block but not previously declared in the main outer block compile, you will not be able to DISPLAY, MOVE or DATATRACE any Pascal procedure, parameter or variable that references the "global but subprogram declared only "Pascal type".

INDEX

F

READER COMMENT SHEET

HP TOOLSET
Reference Manual

32350-90001 January 1984

We welcome your evaluation of this manual. Your comments and suggestions help us to improve our publications. Please explain your answers under Comments, below, and use additional pages if necessary.

Is this manual technically accurate? ☐ **Yes** ☐ **No**

Are the concepts and wording easy to understand? ☐ **Yes** ☐ **No**

Is the format of this manual convenient in size, arrangement, and readability? ☐ **Yes** ☐ **No**

Comments:

This form requires no postage stamp if mailed in the U.S. For locations outside the U.S., your local HP representative will ensure that your comments are forwarded.

FROM: **Date** _____

Name _____

Company _____

Address _____

_____

_____

FOLD FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 1070 CUPERTINO,CA

POSTAGE WILL BE PAID BY ADDRESSEE

Publications Manager
Hewlett-Packard Company
Manufacturing Productivity Division
370 W. Trimble Road
San Jose, CA

FOLD FOLD

Part No. 32350-90001
Printed in U.S.A. January 1984
E0184

HEWLETT PACKARD

SALES & SUPPORT OFFICES

Arranged alphabetically by country

Product Line Sales/Support Key

Key Product Line
A Analytical
CM Components
C Computer Systems Sales only
CH Computer Systems Hardware Sales and Services
CS Computer Systems Software Sales and Services
E Electronic Instruments & Measurement Systems
M Medical Products
MP Medical Products Primary SRO
MS Medical Products Secondary SRO
P Personal Computation Products
*** Sales only for specific product line**
**** Support only for specific product line**

IMPORTANT: These symbols designate general product line capability. They do not insure sales or support availability for all products within a line, at all locations. Contact your local sales office for information regarding locations where HP support is available for specific products.

HP distributors are printed in italics.

HEADQUARTERS OFFICES

If there is no sales office listed for your area, contact one of these headquarters offices.

NORTH/CENTRAL AFRICA

Hewlett-Packard S.A.
7, Rue du Bois-du-Lan
CH-1217 **MEYRIN** 2, Switzerland
Tel: (022) 83 12 12
Telex: 27835 hpse
Cable: HEWPACKSA Geneve

ASIA

Hewlett-Packard Asia Ltd.
6th Floor, Sun Hung Kai Centre
30 Harbour Rd.
G.P.O. Box 795
HONG KONG
Tel: 5-832 3211
After Jan. 1, 1984
47th Floor, China Resources Bldg.
26 Harbour Rd., Wanchai
HONG KONG
Telex: 66678 HEWPA HX
Cable: HEWPACK HONG KONG

CANADA

Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive
MISSISSAUGA, Ontario L4V 1M8
Tel: (416) 678-9430
Telex: 610-492-4246

EASTERN EUROPE

Hewlett-Packard Ges.m.b.h.
Lieblgasse 1
P.O.Box 72
A-1222 **VIENNA,** Austria
Tel: (222) 2365110
Telex: 1 3 4425 HEPA A

NORTHERN EUROPE

Hewlett-Packard S.A.
Uilenstede 475
P.O.Box 999
NL-1180 AZ **AMSTELVEEN**
The Netherlands
Tel: 20 437771

SOUTH EAST EUROPE

Hewlett-Packard S.A.
7, Rue du Bois-du-Lan
CH-1217 **MEYRIN** 2, Switzerland
Tel: (022) 83 12 12
Telex: 27835 hpse
Cable: HEWPACKSA Geneve

OTHER EUROPE

Hewlett-Packard S.A.
P.O. Box
150, Rte du Nant-D'Avril
CH-1217 **MEYRIN** 2, Switzerland
Tel: (022) 83 8111
Telex: 22486 hpsa
Cable: HEWPACKSA Geneve

MEDITERRANEAN AND MIDDLE EAST

Hewlett-Packard S.A.
Mediterranean and Middle East Operations
Atrina Centre
32 Kifissias Ave.
Paradissos-Amarousion, **ATHENS**
Greece
Tel: 682 88 11
Telex: 21-6588 HPAT GR
Cable: HEWPACKSA Athens

EASTERN USA

Hewlett-Packard Co.
4 Choke Cherry Road
ROCKVILLE, MD 20850
Tel: (301) 258-2000

MIDWESTERN USA

Hewlett-Packard Co.
5201 Tollview Drive
ROLLING MEADOWS, IL 60008
Tel: (312) 255-9800

SOUTHERN USA

Hewlett-Packard Co.
2000 South Park Place
P.O. Box 105005
ATLANTA, GA 30348
Tel: (404) 955-1500

WESTERN USA

Hewlett-Packard Co.
3939 Lankershim Blvd.
P.O. Box 3919
LOS ANGELES, CA 91604
Tel: (213) 506-3700

OTHER INTERNATIONAL AREAS

Hewlett-Packard Co.
Intercontinental Headquarters
3495 Deer Creek Road
PALO ALTO, CA 94304
Tel: (415) 857-1501
Telex: 034-8300
Cable: HEWPACK

ANGOLA

Telectra
Empresa Técnica de Equipamentos
R. Barbosa Rodrigues, 41-I DT.
Caixa Postal 6487
LUANDA
Tel: 35515,35516
E,P

ARGENTINA

Hewlett-Packard Argentina S.A.
Avenida Santa Fe 2035
Martinez 1640 **BUENOS AIRES**
Tel: 798-5735, 792-1293
Telex: 17595 BIONAR
Cable: HEWPACKARG
A,E,CH,CS,P

Biotron S.A.C.I.M. e I.
Av Paseo Colon 221, Piso 9
*1399 **BUENOS AIRES***
Tél: 30-4846, 30-1851
Telex: 17595 BIONAR
M

AUSTRALIA

Adelaide, South Australia Office
Hewlett-Packard Australia Ltd.
153 Greenhill Road
PARKSIDE, S.A. 5063
Tel: 272-5911
Telex: 82536
Cable: HEWPARD Adelaide
A*,CH,CM,,E,MS,P

Brisbane, Queensland Office
Hewlett-Packard Australia Ltd.
10 Payne Road
THE GAP, Queensland 4061
Tel: 30-4133
Telex: 42133
Cable: HEWPARD Brisbane
A,CH,CM,E,M,P

Canberra, Australia Capital Territory Office
Hewlett-Packard Australia Ltd.
121 Wollongong Street
FYSHWICK, A.C.T. 2609
Tel: 80 4244
Telex: 62650
Cable: HEWPARD Canberra
CH,CM,E,P

Melbourne, Victoria Office
Hewlett-Packard Australia Ltd.
31-41 Joseph Street
BLACKBURN, Victoria 3130
Tel: 895-2895
Telex: 31-024
Cable: HEWPARD Melbourne
A,CH,CM,CS,E,MS,P

Perth, Western Australia Office
Hewlett-Packard Australia Ltd.
261 Stirling Highway
CLAREMONT, W.A. 6010
Tel: 383-2188
Telex: 93859
Cable: HEWPARD Perth
A,CH,CM,E,MS,P

Sydney, New South Wales Office
Hewlett-Packard Australia Ltd.
17-23 Talavera Road
P.O. Box 308
NORTH RYDE, N.S.W. 2113
Tel: 887-1611
Telex: 21561
Cable: HEWPARD Sydney
A,CH,CM,CS,E,MS,P

AUSTRIA

Hewlett-Packard Ges.m.b.h.
Grottenhofstrasse 94
A-8052 **GRAZ**
Tel: (0316) 291 5 66
Telex: 32375
CH,E

Hewlett-Packard Ges.m.b.h.
Lieblgasse 1
P.O. Box 72
A-1222 **VIENNA**
Tel: (0222) 23 65 11-0
Telex: 134425 HEPA A
A,CH,CM,CS,E,MS,P

BAHRAIN

Green Salon
P.O. Box 557
Manama
BAHRAIN
Tel: 255503-255950
Telex: 84419
P

Wael Pharmacy
P.O. Box 648
BAHRAIN
Tel: 256123
Telex: 8550 WAEL BN
E,C,M

BELGIUM

Hewlett-Packard Belgium S.A./N.V.
Blvd de la Woluwe, 100
Woluwedal
B-1200 **BRUSSELS**
Tel: (02) 762-32-00
Telex: 23-494 paloben bru
A,CH,CM,CS,E,MP,P

BRAZIL

Hewlett-Packard do Brasil I.e.C. Ltda.
Alameda Rio Negro, 750
Alphaville
06400 **BARUERI SP**
Tel: (011) 421.1311
Telex: (011) 33872 HPBR-BR
Cable: HEWPACK Sao Paulo
A,CH,CM,CS,E,M,P

Hewlett-Packard do Brasil I.e.C. Ltda.
Avenida Epitacio Pessoa, 4664
22471 **RIO DE JANEIRO**-RJ
Tel: (02l) 286.0237
Telex: 021-21905 HPBR-BR
Cable: HEWPACK Rio de Janeiro
A,CH,CM,E,MS,P*

ANAMED I.C.E.I. Ltda.
Rua Bage, 103
*04012 **SAO PAULO***
Tel: (011) 570-5726
Telex: 021-21905 HPBR-BR
M

SALES & SUPPORT OFFICES

Arranged alphabetically by country

CANADA

Alberta
Hewlett-Packard (Canada) Ltd.
3030 3rd Avenue N.E.
CALGARY, Alberta T2A 6T7
Tel: (403) 235-3100
A,CH,CM,E*,MS,P*

Hewlett-Packard (Canada) Ltd.
11120A-178th Street
EDMONTON, Alberta T5S 1P2
Tel: (403) 486-6666
A,CH,CM,CS,E,MS,P

British Columbia
Hewlett-Packard (Canada) Ltd.
10691 Shellbridge Way
RICHMOND,
British Columbia V6X 2W7
Tel: (604) 270-2277
Telex: 610-922-5059
A,CH,CM,CS,E*,MS,P*

Manitoba
Hewlett-Packard (Canada) Ltd.
380-550 Century Street
WINNIPEG, Manitoba R3H 0Y1
Tel: (204) 786-6701
A,CH,CM,E,MS,P*

Nova Scotia
Hewlett-Packard (Canada) Ltd.
P.O. Box 931
900 Windmill Road
DARTMOUTH, Nova Scotia B2Y 3Z6
Tel: (902) 469-7820
CH,CM,CS,E*,MS,P*

Ontario
Hewlett-Packard (Canada) Ltd.
3325 N. Service Rd., Unit 6
BURLINGTON, Ontario P3A 2A3
Tel: (416) 335-8644
CS,M*

Hewlett-Packard (Canada) Ltd.
552 Newbold Street
LONDON, Ontario N6E 2S5
Tel: (519) 686-9181
A,CH,CM,E*,MS,P*

Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive
MISSISSAUGA, Ontario L4V 1M8
Tel: (416) 678-9430
A,CH,CM,CS,E,MP,P

Hewlett-Packard (Canada) Ltd.
2670 Queensview Dr.
OTTAWA, Ontario K2B 8K1
Tel: (613) 820-6483
A,CH,CM,CS,E*,MS,P*

Hewlett-Packard (Canada) Ltd.
220 Yorkland Blvd., Unit #11
WILLOWDALE, Ontario M2J 1R5
Tel: (416) 499-9333
CH

Quebec
Hewlett-Packard (Canada) Ltd.
17500 South Service Road
Trans-Canada Highway
KIRKLAND, Quebec H9J 2M5
Tel: (514) 697-4232
A,CH,CM,CS,E,MP,P*

Hewlett-Packard (Canada) Ltd.
Les Galeries du Vallon
2323 Du Versont Nord
STE. FOY, Quebec G1N 4C2
Tel: (418) 687-4570
CH

CHILE

Jorge Calcagni y Cia. Ltda.
Av. Italia 634 Santiago
Casilla 16475
***SANTIAGO** 9*
Tel: 222-0222
Telex: Public Booth 440001
A,CM,E,M

Olympia (Chile) Ltda.
Av. Rodrigo de Araya 1045
Casilla 256-V
***SANTIAGO** 21*
Tel: (02) 22 55 044
Telex: 240-565 OLYMP CL
Cable: Olympiachile Santiagochile
CH,CS,P

CHINA, People's Republic of

China Hewlett-Packard Rep. Office
P.O. Box 418
1A Lane 2, Luchang St.
Beiwei Rd., Xuanwu District
BEIJING
Tel: 33-1947, 33-7426
Telex: 22601 CTSHP CN
Cable: 1920
A,CH,CM,CS,E,P

COLOMBIA

Instrumentación
H. A. Langebaek & Kier S.A.
Carrera 4A No. 52A-26
Apartado Aereo 6287
***BOGOTA** 1, D.E.*
Tel: 212-1466
Telex: 44400 INST CO
Cable: AARIS Bogota
CM,E,M

Casa Humboldt Ltda.
Carrera 14, No. 98-60
Apartado Aereo 51283
***BOGOTA** 1, D.E.*
Tel: 256-1686
Telex: 45403 CCAL CO.
A

COSTA RICA

Cientifica Costarricense S.A.
Avenida 2, Calle 5
San Pedro de Montes de Oca
Apartado 10159
SAN JOSÉ
Tel: 24-38-20, 24-08-19
Telex: 2367 GALGUR CR
CM,E,M

CYPRUS

Telerexa Ltd.
P.O. Box 4809
14C Stassinos Avenue
NICOSIA
Tel: 62698
Telex: 2894 LEVIDO CY
E,M,P

DENMARK

Hewlett-Packard A/S
Datavej 52
DK-3460 **BIRKEROD**
Tel: (02) 81-66-40
Telex: 37409 hpas dk
A,CH,CM,CS,E,MS,P

Hewlett-Packard A/S
Rolighedsvej 32
DK-8240 **RISSKOV,** Aarhus
Tel: (06) 17-60-00
Telex: 37409 hpas dk
CH,E

DOMINICAN REPUBLIC

Microprog S.A.
Juan Tomás Mejía y Cotes No. 60
Arroyo Hondo
SANTO DOMINGO
Tel: 565-6268
Telex: 4510 ARENTA DR (RCA) P

ECUADOR

CYEDE Cia. Ltda.
Avenida Eloy Alfaro 1749
Casilla 6423 CCI
QUITO
Tel: 450-975, 243-052
Telex: 2548 CYEDE ED
CM,E,P

Hospitalar S.A.
Robles 625
Casilla 3590
QUITO
Tel: 545-250, 545-122
Telex: 2485 HOSPTL ED
Cable: HOSPITALAR-Quito
M

EGYPT

International Engineering Associates
24 Hussein Hegazi Street
Kasr-el-Aini
CAIRO
Tel: 23829, 21641
Telex: IEA UN 93830
CH,CS,E,M

EGYPOR
P.O.Box 2558
42 El Zahraa Street
***CAIRO,** Egypt*
Tel: 65 00 21
Telex: 93 337
P

EL SALVADOR

IPESA de El Salvador S.A.
29 Avenida Norte 1216
SAN SALVADOR
Tel: 26-6858, 26-6868
Telex: 20539 IPESASAL
A,CH,CM,CS,E,P

FINLAND

Hewlett-Packard Oy
Revontulentie 7
PL 24
SF-02101 **ESPOO** 10
Tel: (90) 4550211
Telex: 121563 hewpa sf
CH,CM,CS,P

Hewlett-Packard Oy
(Olarinluoma 7)
PL 24
02101 **ESPOO** 10
Tel: (90) 4521022
A,E,MS

Hewlett-Packard Oy
Aatoksenkatv 10-C
SF-40720-72 **JYVASKYLA**
Tel: (941) 216318
CH

Hewlett-Packard Oy
Kainvuntie 1-C
SF-90140-14 **OULU**
Tel: (981) 338785
CH

FRANCE

Hewlett-Packard France
Z.I. Mercure B
Rue Berthelot
F-13763 Les Milles Cedex
AIX-EN-PROVENCE
Tel: 16 (42) 59-41-02
Telex: 410770F
A,CH,E,MS,P*

Hewlett-Packard France
64, rue Marchand Saillant
F-61000 **ALENCON**
Tel: 16 (33) 29 04 42

Hewlett-Packard France
Boite Postale 503
F-25026 **BESANCON**
28 rue de la Republique
F-25000 **BESANCON**
Tel: 16 (81) 83-16-22
CH,M

Hewlett-Packard France
13, Place Napoleon III
F-29000 **BREST**
Tel: 16 (98) 03-38-35

Hewlett-Packard France
Chemin des Mouilles
Boite Postale 162
F-69130 **ECULLY** Cedex (Lyon)
Tel: 16 (78) 833-81-25
Telex: 310617F
A,CH,CS,E,MP

Hewlett-Packard France
Tour Lorraine
Boulevard de France
F-91035 **EVRY** Cedex
Tel: 16 6 077-96-60
Telex: 692315F
E

Hewlett-Packard France
Parc d'Activite du Bois Briard
Ave. du Lac
F-91040 **EVRY** Cedex
Tel: 16 6 077-8383
Telex: 692315F
E

Hewlett-Packard France
5, avenue Raymond Chanas
F-38320 **EYBENS** (Grenoble)
Tel: 16 (76) 25-81-41
Telex: 980124 HP GRENOB EYBE
CH

Hewlett-Packard France
Centre d'Affaire Paris-Nord
Bâtiment Ampère 5 étage
Rue de la Commune de Paris
Boite Postale 300
F-93153 **LE BLANC MESNIL**
Tel: 16 (1) 865-44-52
Telex: 211032F
CH,CS,E,MS

Hewlett-Packard France
Parc d'Activités Cadera
Quartier Jean Mermoz
Avenue du Président JF Kennedy
F-33700 **MERIGNAC** (Bordeaux)
Tel: 16 (56) 34-00-84
Telex: 550105F
CH,E,MS

Hewlett-Packard France
Immueble "Les 3 B"
Nouveau Chemin de la Garde
ZAC de Bois Briand
F-44085 **NANTES** Cedex
Tel: 16 (40) 50-32-22
CH**

FRANCE (Cont'd)

Hewlett-Packard France
125, rue du Faubourg Bannier
F-45000 **ORLEANS**
Tel: 16 (38) 68 01 63

Hewlett-Packard France
Zone Industrielle de Courtaboeuf
Avenue des Tropiques
F-91947 Les Ulis Cedex **ORSAY**
Tel: (6) 907-78-25
Telex: 600048F
A,CH,CM,CS,E,MP,P

Hewlett-Packard France
Paris Porte-Maillot
15, Avenue de L'Amiral Bruix
F-75782 **PARIS** CEDEX 16
Tel: 16 (1) 502-12-20
Telex: 613663F
CH,MS,P

Hewlett-Packard France
124, Boulevard Tourasse
F-64000 **PAU**
Tel: 16 (59) 80 38 02

Hewlett-Packard France
2 Allée de la Bourgonnette
F-35100 **RENNES**
Tel: 16 (99) 51-42-44
Telex: 740912F
CH,CM,E,MS,P*

Hewlett-Packard France
98 Avenue de Bretagne
F-76100 **ROUEN**
Tel: 16 (35) 63-57-66
CH**,CS

Hewlett-Packard France
4 Rue Thomas Mann
Boite Postale 56
F-67033 **STRASBOURG** Cedex
Tel: 16 (88) 28-56-46
Telex: 890141F
CH,E,MS,P*

Hewlett-Packard France
Le Péripole
20, Chemin du Pigeonnier de la Cépière
F-31083 **TOULOUSE** Cedex
Tel: 16 (61) 40-11-12
Telex: 531639F
A,CH,CS,E,P*

Hewlett-Packard France
9, rue Baudin
F-26000 **VALENCE**
Tel: 16 (75) 42 76 16

Hewlett-Packard France
Carolor
ZAC de Bois Briand
F-57640 **VIGY** (Metz)
Tel: 16 (8) 771 20 22
CH

Hewlett-Packard France
Immeuble Péricentre
F-59658 **VILLENEUVE D'ASCQ** Cedex
Tel: 16 (20) 91-41-25
Telex: 160124F
CH,E,MS,P*

GERMAN FEDERAL REPUBLIC

Hewlett-Packard GmbH
Geschäftsstelle
Keithstrasse 2-4
D-1000 **BERLIN** 30
Tel: (030) 24-90-86
Telex: 018 3405 hpbln d
A,CH,E,M,P

Hewlett-Packard GmbH
Geschäftsstelle
Herrenberger Strasse 130
D-7030 **BÖBLINGEN**
Tel: (7031) 14-0
Telex:
A,CH,CM,CS,E,MP,P

Hewlett-Packard GmbH
Geschäftsstelle
Emanuel-Leutze-Strasse 1
D-4000 **DUSSELDORF**
Tel: (0211) 5971-1
Telex: 085/86 533 hpdd d
A,CH,CS,E,MS,P

Hewlett-Packard GmbH
Geschäftsstelle
Schleefstr. 28a
D-4600 **DORTMUND**-Aplerbeck
Tel: (0231) 45001

Hewlett-Packard GmbH
Vertriebszentrale Frankfurt
Berner Strasse 117
Postfach 560 140
D-6000 **FRANKFURT** 56
Tel: (0611) 50-04-1
Telex: 04 13249 hpffm d
A,CH,CM,CS,E,MP,P

Hewlett-Packard GmbH
Geschäftsstelle
Aussenstelle Bad Homburg
Louisenstrasse 115
D-6380 **BAD HOMBURG**
Tel: (06172) 109-0

Hewlett-Packard GmbH
Geschäftsstelle
Kapstadtring 5
D-2000 **HAMBURG** 60
Tel: (040) 63804-1
Telex: 021 63 032 hphh d
A,CH,CS,E,MS,P

Hewlett-Packard GmbH
Geschäftsstelle
Heidering 37-39
D-3000 **HANNOVER** 61
Tel: (0511) 5706-0
Telex: 092 3259
A,CH,CM,E,MS,P

Hewlett-Packard GmbH
Geschäftsstelle
Rosslauer Weg 2-4
D-6800 **MANNHEIM**
Tel: (0621) 70050
Telex: 0462105
A,C,E

Hewlett-Packard GmbH
Geschäftsstelle
Messerschmittstrasse 7
D-7910 **NEU ULM**
Tel: 0731-70241
Telex: 0712816 HP ULM-D
A,C,E*

Hewlett-Packard GmbH
Geschäftsstelle
Ehhericherstr. 13
D-8500 **NÜRNBERG** 10
Tel: (0911) 5205-0
Telex: 0623 860
CH,CM,E,MS,P

Hewlett-Packard GmbH
Geschäftsstelle
Eschenstrasse 5
D-8028 **TAUFKIRCHEN**
Tel: (089) 6117-1
Telex: 0524985
A,CH,CM,E,MS,P

GREAT BRITAIN

See United Kingdom

GREECE

Kostas Karaynnis S.A.
8 Omirou Street
***ATHENS** 133*
Tel: 32 30 303, 32 37 371
Telex: 215962 RKAR GR
A,CH,CM,CS,E,M,P

PLAISIO S.A.
G. Gerardos
24 Stournara Street
ATHENS
Tel: 36-11-160
Telex: 221871
P

GUATEMALA

IPESA
Avenida Reforma 3-48, Zona 9
GUATEMALA CITY
Tel: 316627, 314786
Telex: 4192 TELTRO GU
A,CH,CM,CS,E,M,P

HONG KONG

Hewlett-Packard Hong Kong, Ltd.
G.P.O. Box 795
5th Floor, Sun Hung Kai Centre
30 Harbour Road
HONG KONG
Tel: 5-8323211
Telex: 66678 HEWPA HX
Cable: HEWPACK HONG KONG
E,CH,CS,P

CET Ltd.
1402 Tung Wah Mansion
199-203 Hennessy Rd.
*Wanchia, **HONG KONG***
Tel: 5-729376
Telex: 85148 CET HX
CM

Schmidt & Co. (Hong Kong) Ltd.
Wing On Centre, 28th Floor
Connaught Road, C.
HONG KONG
Tel: 5-455644
Telex: 74766 SCHMX HX
A,M

ICELAND

Elding Trading Company Inc.
Hafnarnvoli-Tryggvagotu
P.O. Box 895
IS-REYKJAVIK
Tel: 1-58-20, 1-63-03
M

INDIA

Computer products are sold through Blue Star Ltd. All computer repairs and maintenance service is done through Computer Maintenance Corp.

Blue Star Ltd.
Sabri Complex II Floor
24 Residency Rd.
***BANGALORE** 560 025*
Tel: 55660
Telex: 0845-430
Cable: BLUESTAR
A,CH,CM,CS*,E*

Blue Star Ltd.
Band Box House
Prabhadevi
***BOMBAY** 400 025*
Tel: 422-3101
Telex: 011-3751
Cable: BLUESTAR
A,M

Blue Star Ltd.
Sahas
414/2 Vir Savarkar Marg
Prabhadevi
***BOMBAY** 400 025*
Tel: 422-6155
Telex: 011-4093
Cable: FROSTBLUE
A,CH,CM,CS*,E,M*

Blue Star Ltd.
Kalyan, 19 Vishwas Colony
*Alkapuri, **BORODA**, 390 005*
Tel: 65235
Cable: BLUE STAR
A

Blue Star Ltd.
7 Hare Street
***CALCUTTA** 700 001*
Tel: 12-01-31
Telex: 021-7655
Cable: BLUESTAR
A,M

Blue Star Ltd.
133 Kodambakkam High Road
***MADRAS** 600 034*
Tel: 82057
Telex: 041-379
Cable: BLUESTAR
A,M

Blue Star Ltd.
Bhandari House, 7th/8th Floors
91 Nehru Place
***NEW DELHI** 110 024*
Tel: 682547
Telex: 031-2463
Cable: BLUESTAR
A,CH,CM,CS*,E,M*

Blue Star Ltd.
15/16:C Wellesley Rd.
***PUNE** 411 011*
Tel: 22775
Cable: BLUE STAR
A

Blue Star Ltd.
2-2-47/1108 Bolarum Rd.
***SECUNDERABAD** 500 003*
Tel: 72057
Telex: 0155-459
Cable: BLUEFROST
A,E

Blue Star Ltd.
T.C. 7/603 Poornima
Maruthankuzhi
***TRIVANDRUM** 695 013*
Tel: 65799
Telex: 0884-259
Cable: BLUESTAR
E

Computer Maintenance Corporation Ltd.
115, Sarojini Devi Road
***SECUNDERABAD** 500 003*
Tel: 310-184, 345-774
Telex: 031-2960
*CH***

SALES & SUPPORT OFFICES

Arranged alphabetically by country

INDONESIA

BERCA Indonesia P.T.
P.O.Box 496/Jkt.
Jl. Abdul Muis 62
JAKARTA
Tel: 21-373009
Telex: 46748 BERSAL IA
Cable: BERSAL JAKARTA
P

BERCA Indonesia P.T.
P.O.Box 2497/Jkt
Antara Bldg., 17th Floor
Jl. Medan Merdeka Selatan 17
JAKARTA-PUSAT
Tel: 21-344-181
Telex: BERSAL IA
A,CS,E,M

BERCA Indonesia P.T.
P.O. Box 174/SBY.
Jl. Kutei No. 11
SURABAYA
Tel: 68172
Telex: 31146 BERSAL SB
Cable: BERSAL-SURABAYA
A,E,M,P*

IRAQ

Hewlett-Packard Trading S.A.
Service Operation
Al Mansoor City 9B/3/7
BAGHDAD
Tel: 551-49-73
Telex: 212-455 HEPAIRAQ IK
CH,CS

IRELAND

Hewlett-Packard Ireland Ltd.
82/83 Lower Leeson Street
DUBLIN 2
Tel: 0001 608800
Telex: 30439
A,CH,CM,CS,E,M,P

Cardiac Services Ltd.
Kilmore Road
Artane
***DUBLIN** 5*
Tel: (01) 351820
Telex: 30439
M

ISRAEL

Eldan Electronic Instrument Ltd.
P.O.Box 1270
***JERUSALEM** 91000*
16, Ohaliav St.
***JERUSALEM** 94467*
Tel: 533 221, 553 242
Telex: 25231 AB/PAKRD IL
A

Electronics Engineering Division
Motorola Israel Ltd.
16 Kremenetski Street
P.O. Box 25016
TEL-AVIV 67899
Tel: 3 88 388
Telex: 33569 Motil IL
Cable: BASTEL Tel-Aviv
CH,CM,CS,E,M,P

ITALY

Hewlett-Packard Italiana S.p.A
Traversa 99C
Via Giulio Petroni, 19
I-70124 **BARI**
Tel: (080) 41-07-44
M

Hewlett-Packard Italiana S.p.A.
Via Martin Luther King, 38/III
I-40132 **BOLOGNA**
Tel: (051) 402394
Telex: 511630
CH,E,MS

Hewlett-Packard Italiana S.p.A.
Via Principe Nicola 43G/C
I-95126 **CATANIA**
Tel: (095) 37-10-87
Telex: 970291
C,P

Hewlett-Packard Italiana S.p.A.
Via G. Di Vittorio 9
I-20063 **CERNUSCO SUL NAVIGLIO**
(Milano)
Tel: (02) 923691
Telex: 334632
A,CH,CM,CS,E,MP,P

Hewlett-Packard Italiana S.p.A.
Via C. Colombo 49
I-20090 **TREZZANO SUL NAVIGLIO**
(Milano)
Tel: (02) 4459041
Telex: 322116
C,M

Hewlett-Packard Italiana S.p.A.
Via Nuova San Rocco a
Capodimonte, 62/A
I-80131 **NAPOLI**
Tel: (081) 7413544
Telex: 710698
A,CH,E

Hewlett-Packard Italiana S.p.A.
Viale G. Modugno 33
I-16156 **GENOVA PEGLI**
Tel: (010) 68-37-07
Telex: 215238
E,C

Hewlett-Packard Italiana S.p.A.
Via Pelizzo 15
I-35128 **PADOVA**
Tel: (049) 664888
Telex: 430315
A,CH,E,MS

Hewlett-Packard Italiana S.p.A.
Viale C. Pavese 340
I-00144 **ROMA EUR**
Tel: (06) 54831
Telex: 610514
A,CH,CM,CS,E,MS,P*

Hewlett-Packard Italiana S.p.A.
Via di Casellina 57/C
I-50018 **SCANDICCI-FIRENZE**
Tel: (055) 753863

Hewlett-Packard Italiana S.p.A.
Corso Svizzera, 185
I-10144 **TORINO**
Tel: (011) 74 4044
Telex: 221079
CH,E

JAPAN

Yokogawa-Hewlett-Packard Ltd.
152-1, Onna
ATSUGI, Kanagawa, 243
Tel: (0462) 28-0451
CM,C*,E

Yokogawa-Helwett-Packard Ltd.
Meiji-Seimei Bldg. 6F
3-1 Hon Chiba-Cho
CHIBA, 280
Tel: 472 25 7701
E,CH,CS

Yokogawa-Hewlett-Packard Ltd.
Yasuda-Seimei Hiroshima Bldg.
6-11, Hon-dori, Naka-ku
HIROSHIMA, 730
Tel: 82-241-0611

Yokogawa-Hewlett-Packard Ltd.
Towa Building
2-3, Kaigan-dori, 2 Chome Chuo-ku
KOBE, 650
Tel: (078) 392-4791
C,E

Yokogawa-Hewlett-Packard Ltd.
Kumagaya Asahi 82 Bldg
3-4 Tsukuba
KUMAGAYA, Saitama 360
Tel: (0485) 24-6563
CH,CM,E

Yokogawa-Hewlett-Packard Ltd.
Asahi Shinbun Daiichi Seimei Bldg.
4-7, Hanabata-cho
KUMAMOTO,860
Tel: (0963) 54-7311
CH,E

Yokogawa-Hewlett-Packard Ltd.
Shin-Kyoto Center Bldg.
614, Higashi-Shiokoji-cho
Karasuma-Nishiiru
Shiokoji-dori, Shimogyo-ku
KYOTO, 600
Tel: 075-343-0921
CH,E

Yokogawa-Hewlett-Packard Ltd.
Mito Mitsui Bldg
4-73, Sanno-maru, 1 Chome
MITO, Ibaraki 310
Tel: (0292) 25-7470
CH,CM,E

Yokogawa-Hewlett-Packard Ltd.
Sumitomo Seimei 14-9 Bldg.
Meieki-Minami, 2 Chome
Nakamura-ku
NAGOYA, 450
Tel: (052) 571-5171
CH,CM,CS,E,MS

Yokogawa-Hewlett-Packard Ltd.
Chuo Bldg.,
4-20 Nishinakajima, 5 Chome
Yodogawa-ku
OSAKA, 532
Tel: (06) 304-6021
Telex: YHPOSA 523-3624
A,CH,CM,CS,E,MP,P*

Yokogawa-Hewlett-Packard Ltd.
27-15, Yabe, 1 Chome
SAGAMIHARA Kanagawa, 229
Tel: 0427 59-1311

Yokogawa-Hewlett-Packard Ltd.
Daiichi Seimei Bldg.
7-1, Nishi Shinjuku, 2 Chome
Shinjuku-ku,**TOKYO** 160
Tel: 03-348-4611
CH,E

Yokogawa-Hewlett-Packard Ltd.
29-21 Takaido-Higashi, 3 Chome
Suginami-ku **TOKYO** 168
Tel: (03) 331-611
Telex: 232-2024 YHPTOK
A,CH,CM,CS,E,MP,P*

Yokogawa-Hewlett-Packard Ltd.
Daiichi Asano Building
2-8, Odori, 5 Chome
UTSUNOMIYA, Tochigi 320
Tel: (0286) 25-7155
CH,CS,E

Yokogawa-Hewlett-Packard Ltd.
Yasuda Seimei Nishiguchi Bldg.
30-4 Tsuruya-cho, 3 Chome
YOKOHAMA 221
Tel: (045) 312-1252
CH,CM,E

JORDAN

Mouasher Cousins Company
P.O. Box 1387
AMMAN
Tel: 24907, 39907
Telex: 21456 SABCO JO
CH,E,M,P

KENYA

ADCOM Ltd., Inc., Kenya
P.O.Box 30070
NAIROBI
Tel: 331955
Telex: 22639
E,M

KOREA

Samsung Electronics HP Division
12 Fl. Kinam Bldg.
San 75-31, Yeoksam-Dong
Kangnam-Ku
Yeongdong P.O. Box 72
SEOUL
Tel: 555-7555, 555-5447
Telex: K27364 SAMSAN
A,CH,CM,CS,E,M,P

KUWAIT

Al-Khaldiya Trading & Contracting
P.O. Box 830 Safat
KUWAIT
Tel: 42-4910, 41-1726
Telex: 22481 Areeg kt
CH,E,M

Photo & Cine Equipment
P.O. Box 270 Safat
KUWAIT
Tel: 42-2846, 42-3801
Telex: 22247 Matin kt
P

LEBANON

G.M. Dolmadjian
Achrafieh
P.O. Box 165.167
BEIRUT
Tel: 290293
*MP***

Computer Information Systems
P.O. Box 11-6274
BEIRUT
Tel: 89 40 73
Telex: 22259
C

LUXEMBOURG

Hewlett-Packard Belgium S.A./N.V.
Blvd de la Woluwe, 100
Woluwedal
B-1200 **BRUSSELS**
Tel: (02) 762-32-00
Telex: 23-494 paloben bru
A,CH,CM,CS,E,MP,P

MALAYSIA

Hewlett-Packard Sales (Malaysia)
Sdn. Bhd.
1st Floor, Bangunan British
American
Jalan Semantan, Damansara Heights
KUALA LUMPUR 23-03
Tel: 943022
Telex: MA31011
A,CH,E,M,P*

MAYLAYSIA (Cont'd)
Protel Engineering
P.O.Box 1917
Lot 6624, Section 64
23/4 Pending Road
Kuching, ***SARAWAK***
Tel: 36299
Telex: MA 70904 PROMAL
Cable: PROTELENG
A,E,M

MALTA
Philip Toledo Ltd.
Notabile Rd.
MRIEHEL
Tel: 447 47, 455 66
Telex: Media MW 649
E,P

MEXICO
Hewlett-Packard Mexicana, S.A. de C.V.
Av. Periferico Sur No. 6501
Tepepan, Xochimilco
16020 **MEXICO D.F.**
Tel: 6-76-46-00
Telex: 17-74-507 HEWPACK MEX
A,CH,CS,E,MS,P

Hewlett-Packard Mexicana, S.A. de C.V.
Ave. Colonia del Valle 409
Col. del Valle
Municipio de Garza Garcia
MONTERREY, Nuevo Leon
Tel: 78 42 41
Telex: 038 410
CH

ECISA
José Vasconcelos No. 218
Col. Condesa Deleg. Cuauhtémoc
MEXICO *D.F. 06140*
Tel: 553-1206
Telex: 17-72755 ECE ME
M

MOROCCO
Dolbeau
81 rue Karatchi
CASABLANCA
Tel: 3041-82, 3068-38
Telex: 23051, 22822
E

Gerep
2 rue d'Agadir
Boite Postale 156
CASABLANCA
Tel: 272093, 272095
Telex: 23 739
P

NETHERLANDS
Hewlett-Packard Nederland B.V.
Van Heuven Goedhartlaan 121
NL 1181KK **AMSTELVEEN**
P.O. Box 667
NL1180 AR **AMSTELVEEN**
Tel: (020) 47-20-21
Telex: 13 216 HEPA NL
A,CH,CM,CS,E,MP,P

Hewlett-Packard Nederland B.V.
Bongerd 2
NL 2906VK **CAPELLE A/D IJSSEL**
P.O. Box 41
NL 2900AA **CAPELLE A/D IJSSEL**
Tel: (10) 51-64-44
Telex: 21261 HEPAC NL
A,CH,CS,E

Hewlett-Packard Nederland B.V.
Pastoor Petersstraat 134-136
NL 5612 LV **EINDHOVEN**
P.O. Box 2342
NL 5600 CH **EINDHOVEN**
Tel: (040) 326911
Telex: 51484 hepae nl
A,CH**,E,M

NEW ZEALAND
Hewlett-Packard (N.Z.) Ltd.
5 Owens Road
P.O. Box 26-189
Epsom, **AUCKLAND**
Tel: 687-159
Cable: HEWPACK Auckland
CH,CM,E,P*

Hewlett-Packard (N.Z.) Ltd.
4-12 Cruickshank Street
Kilbirnie, **WELLINGTON 3**
P.O. Box 9443
Courtenay Place, **WELLINGTON 3**
Tel: 877-199
Cable: HEWPACK Wellington
CH,CM,E,P

Northrop Instruments & Systems Ltd.
369 Khyber Pass Road
P.O. Box 8602
AUCKLAND
Tel: 794-091
Telex: 60605
A,M

Northrop Instruments & Systems Ltd.
110 Mandeville St.
P.O. Box 8388
CHRISTCHURCH
Tel: 486-928
Telex: 4203
A,M

Northrop Instruments & Systems Ltd.
Sturdee House
85-87 Ghuznee Street
P.O. Box 2406
WELLINGTON
Tel: 850-091
Telex: NZ 3380
A,M

NORTHERN IRELAND
See United Kingdom

NORWAY
Hewlett-Packard Norge A/S
Folke Bernadottes vei 50
P.O. Box 3558
N-5033 **FYLLINGSDALEN** (Bergen)
Tel: 0047/5/16 55 40
Telex: 16621 hpnas n
CH,CS,E,MS

Hewlett-Packard Norge A/S
Österndalen 16-18
P.O. Box 34
N-1345 **ÖSTERÅS**
Tel: 0047/2/17 11 80
Telex: 16621 hpnas n
A,CH,CM,CS,E,M,P

OMAN
Khimjil Ramdas
P.O. Box 19
MUSCAT
Tel: 722225, 745601
Telex: 3289 BROKER MB MUSCAT
P

Suhail & Saud Bahwan
P.O.Box 169
MUSCAT
Tel: 734 201-3
Telex: 3274 BAHWAN MB

PAKISTAN
Mushko & Company Ltd.
1-B, Street 43
Sector F-8/1
ISLAMABAD
Tel: 51071
Cable: FEMUS Rawalpindi
A,E,M

Mushko & Company Ltd.
Oosman Chambers
Abdullah Haroon Road
KARACHI *0302*
Tel: 524131, 524132
Telex: 2894 MUSKO PK
Cable: COOPERATOR Karachi
*A,E,M,P**

PANAMA
Electrónico Balboa, S.A.
Calle Samuel Lewis, Ed. Alfa
Apartado 4929
PANAMA 5
Tel: 63-6613, 63-6748
Telex: 3483 ELECTRON PG
A,CM,E,M,P

PERU
Cía Electro Médica S.A.
Los Flamencos 145, San Isidro
Casilla 1030
LIMA *1*
Tel: 41-4325, 41-3703
Telex: Pub. Booth 25306
CM,E,M,P

PHILIPPINES
The Online Advanced Systems Corporation
Rico House, Amorsolo Cor. Herrera Street
Legaspi Village, Makati
P.O. Box 1510
Metro ***MANILA***
Tel: 85-35-81, 85-34-91, 85-32-21
Telex: 3274 ONLINE
A,CH,CS,E,M

Electronic Specialists and Proponents Inc.
690-B Epifanio de los Santos Avenue
Cubao, ***QUEZON CITY***
P.O. Box 2649 Manila
Tel: 98-96-81, 98-96-82, 98-96-83
Telex: 40018, 42000 ITT GLOBE MACKAY BOOTH
P

PORTUGAL
Mundinter
Intercambio Mundial de Comércio S.A.R.L.
P.O. Box 2761
Av. Antonio Augusto de Aguiar 138
P-LISBON
Tel: (19) 53-21-31, 53-21-37
Telex: 16691 munter p
M

Soquimica
Av. da Liberdade, 220-2
1298 ***LISBOA*** *Codex*
Tel: 56 21 81/2/3
Telex: 13316 SABASA
P

Telectra-Empresa Técnica de Equipmentos Eléctricos S.A.R.L.
Rua Rodrigo da Fonseca 103
P.O. Box 2531
P-LISBON *1*
Tel: (19) 68-60-72
Telex: 12598
CH,CS,E,P

PUERTO RICO
Hewlett-Packard Puerto Rico
Ave. Muñoz Rivera #101
Esq. Calle Ochoa
HATO REY, Puerto Rico 00918
Tel: (809) 754-7800

Hewlett-Packard Puerto Rico
Calle 272 Edificio 203
Urb. Country Club
RIO PIEDRAS, Puerto Rico
P.O. Box 4407
CAROLINA, Puerto Rico 00628
Tel: (809) 762-7255
A,CH,CS

QATAR
Computearbia
P.O. Box 2750
DOHA
Tel: 883555
Telex: 4806 CHPARB
P

Eastern Technical Services
P.O.Box 4747
DOHA
Tel: 329 993
Telex: 4156 EASTEC DH

Nasser Trading & Contracting
P.O.Box 1563
DOHA
Tel: 22170, 23539
Telex: 4439 NASSER DH
M

SAUDI ARABIA
Modern Electronic Establishment
Hewlett-Packard Division
P.O. Box 22015
Thuobah
AL-KHOBAR
Tel: 895-1760, 895-1764
Telex: 671 106 HPMEEK SJ
Cable: ELECTA AL-KHOBAR
CH,CS,E,M

Modern Electronic Establishment
Hewlett-Packard Division
P.O. Box 1228
Redec Plaza, 6th Floor
JEDDAH
Tel: 644 38 48
Telex: 4027 12 FARNAS SJ
Cable: ELECTA JEDDAH
CH,CS,E,M

Modern Electronic Establishment
Hewlett-Packard Division
P.O.Box 22015
RIYADH
Tel: 491-97 15, 491-63 87
Telex: 202049 MEERYD SJ
CH,CS,E,M

Abdul Ghani El Ajou
P.O. Box 78
RIYADH
Tel: 40 41 717
Telex: 200 932 EL AJOU
P

SCOTLAND
See United Kingdom

SINGAPORE
Hewlett-Packard Singapore (Sales) Pte. Ltd.
#08-00 Inchcape House
450-2 Alexandra Road
P.O. Box 58 Alexandra Rd. Post Office
SINGAPORE, 9115
Tel: 631788
Telex: HPSGSO RS 34209
Cable: HEWPACK, Singapore
A,CH,CS,E,MS,P

SALES & SUPPORT OFFICES

Arranged alphabetically by country

SINGAPORE (Cont'd)

Dynamar International Ltd.
Unit 05-11 Block 6
Kolam Ayer Industrial Estate
***SINGAPORE** 1334*
Tel: 747-6188
Telex: RS 26283
CM

SOUTH AFRICA

Hewlett-Packard So Africa (Pty.) Ltd.
P.O. Box 120
Howard Place **CAPE PROVINCE** 7450
Pine Park Center, Forest Drive,
Pinelands
CAPE PROVINCE 7405
Tel: 53-7954
Telex: 57-20006
A,CH,CM,E,MS,P

Hewlett-Packard So Africa (Pty.) Ltd.
P.O. Box 37099
92 Overport Drive
DURBAN 4067
Tel: 28-4178, 28-4179, 28-4110
Telex: 6-22954
CH,CM

Hewlett-Packard So Africa (Pty.) Ltd.
6 Linton Arcade
511 Cape Road
Linton Grange
PORT ELIZABETH 6000
Tel: 041-302148
CH

Hewlett-Packard So Africa (Pty.) Ltd.
P.O.Box 33345
Glenstantia 0010 **TRANSVAAL**
1st Floor East
Constantia Park Ridge Shopping
Centre
Constantia Park
PRETORIA
Tel: 982043
Telex: 32163
CH,E

Hewlett-Packard So Africa (Pty.) Ltd.
Private Bag Wendywood
SANDTON 2144
Tel: 802-5111, 802-5125
Telex: 4-20877
Cable: HEWPACK Johannesburg
A,CH,CM,CS,E,MS,P

SPAIN

Hewlett-Packard Española S.A.
Calle Entenza, 321
E-BARCELONA 29
Tel: 322.24.51, 321.73.54
Telex: 52603 hpbee
A,CH,CS,E,MS,P

Hewlett-Packard Española S.A.
Calle San Vicente S/No
Edificio Albia II
E-BILBAO 1
Tel: 423.83.06
A,CH,E,MS

Hewlett-Packard Española S.A.
Crta. de la Coruña, Km. 16, 400
Las Rozas
E-MADRID
Tel: (1) 637.00.11
CH,CS,M

Hewlett-Packard Española S.A.
Avda. S. Francisco Javier, S/no
Planta 10. Edificio Sevilla 2,
E-SEVILLA 5
Tel: 64.44.54
Telex: 72933
A,CS,MS,P

Hewlett-Packard Española S.A.
Calle Ramon Gordillo, 1 (Entlo.3)
E-VALENCIA 10
Tel: 361-1354
CH,P

SWEDEN

Hewlett-Packard Sverige AB
Sunnanvagen 14K
S-22226 **LUND**
Tel: (046) 13-69-79
Telex: (854) 17886 (via Spånga
office)
CH

Hewlett-Packard Sverige AB
Östra Tullgatan 3
S-21128 **MALMÖ**
Tel: (040) 70270
Telex: (854) 17886 (via Spånga
office)

Hewlett-Packard Sverige AB
Västra Vintergatan 9
S-70344 **ÖREBRO**
Tel: (19) 10-48-80
Telex: (854) 17886 (via Spånga
office)
CH

Hewlett-Packard Sverige AB
Skalholtsgatan 9, Kista
Box 19
S-16393 **SPÅNGA**
Tel: (08) 750-2000
Telex: (854) 17886
Telefax: (08) 7527781
A,CH,CM,CS,E,MS,P

Hewlett-Packard Sverige AB
Frötallisgatan 30
S-42132 **VÄSTRA-FRÖLUNDA**
Tel: (031) 49-09-50
Telex: (854) 17886 (via Spånga
office)
CH,E,P

SWITZERLAND

Hewlett-Packard (Schweiz) AG
Clarastrasse 12
CH-4058 **BASEL**
Tel: (61) 33-59-20
A

Hewlett-Packard (Schweiz) AG
7, rue du Bois-du-Lan
Case Postale 365
CH-1217 **MEYRIN** 2
Tel: (0041) 22-83-11-11
Telex:27333 HPAG CH
CH,CM,CS

Hewlett-Packard (Schweiz) AG
Allmend 2
CH-8967 **WIDEN**
Tel: (0041) 57 31 21 11
Telex: 53933 hpag ch
Cable: HPAG CH
A,CH,CM,CS,E,MS,P

SYRIA

General Electronic Inc.
Nuri Basha Ahnaf Ebn Kays Street
P.O. Box 5781
DAMASCUS
Tel: 33-24-87
Telex: 411 215
Cable: ELECTROBOR DAMASCUS
E

Middle East Electronics
P.O.Box 2308
Abu Rumnaneh
DAMASCUS
Tel: 33 4 5 92
Telex: 411 304
M

TAIWAN

Hewlett-Packard Far East Ltd.
Kaohsiung Office
2/F 68-2, Chung Cheng 3rd Road
KAOHSIUNG
Tel: (07) 241-2318
CH,CS,E

Hewlett-Packard Far East Ltd.
Taiwan Branch
8th Floor
337 Fu Hsing North Road
TAIPEI
Tel: (02) 712-0404
Telex: 24439 HEWPACK
Cable:HEWPACK Taipei
A,CH,CM,CS,E,M,P

Ing Lih Trading Co.
3rd Floor, 7 Jen-Ai Road, Sec. 2
TAIPEI 100
Tel: (02) 3948191
Cable: INGLIH TAIPEI
A

THAILAND

Unimesa
30 Patpong Ave., Suriwong
***BANGKOK** 5*
Tel: 235-5727
Telex: 84439 Simonco TH
Cable: UNIMESA Bangkok
A,CH,CS,E,M

Bangkok Business Equipment Ltd.
5/5-6 Dejo Road
BANGKOK
Tel: 234-8670, 234-8671
Telex: 87669-BEQUIPT TH
Cable: BUSIQUIPT Bangkok
P

TRINIDAD & TOBAGO

Caribbean Telecoms Ltd.
50/A Jerningham Avenue
P.O. Box 732
PORT-OF-SPAIN
Tel: 62-44213, 62-44214
Telex: 235,272 HUGCO WG
CM,E,M,P

TUNISIA

Tunisie Electronique
31 Avenue de la Liberte
TUNIS
Tel: 280-144
E,P

Corema
1 ter. Av. de Carthage
TUNIS
Tel: 253-821
Telex: 12319 CABAM TN
M

TURKEY

Teknim Company Ltd.
Iran Caddesi No. 7
*Kavaklidere, **ANKARA***
Tel: 275800
Telex: 42155 TKNM TR
E

E.M.A.
Medina Eldem Sokak No.41/6
Yuksel Caddesi
ANKARA
Tel: 175 622
Telex: 42 591
M

UNITED ARAB EMIRATES

Emitac Ltd.
P.O. Box 2711
ABU DHABI
Tel: 82 04 19-20
Cable: EMITAC ABUDHABI

Emitac Ltd.
P.O. Box 1641
SHARJAH
Tel: 591 181
Telex: 68136 Emitac Sh
CH,CS,E,M,P

UNITED KINGDOM

GREAT BRITAIN

Hewlett-Packard Ltd.
Trafalgar House
Navigation Road
ALTRINCHAM
Cheshire WA14 1NU
Tel: 061 928 6422
Telex: 668068
A,CH,CS,E,M,MS,P

Hewlett-Packard Ltd.
Elstree House, Elstree Way
BOREHAMWOOD, Herts WD6 1SG
Tel: 01 207 5000
Telex: 8952716
E,CH,CS,P

Hewlett-Packard Ltd.
Oakfield House, Oakfield Grove
Clifton **BRISTOL,** Avon BS8 2BN
Tel: 0272 736806
Telex: 444302
CH,CS,E,P

Hewlett-Packard Ltd.
Bridewell House
Bridewell Place
LONDON EC4V 6BS
Tel: 01 583 6565
Telex: 298163
CH,CS,P

Hewlett-Packard Ltd.
Fourier House
257-263 High Street
LONDON COLNEY
Herts. AL2 1HA, St. Albans
Tel: 0727 24400
Telex: 1-8952716
CH,CS

Hewlett-Packard Ltd.
Pontefract Road
NORMANTON, West Yorkshire WF6 1RN
Tel: 0924 895566
Telex: 557355
CH,CS,P

Hewlett-Packard Ltd.
The Quadrangle
106-118 Station Road
REDHILL, Surrey RH1 1PS
Tel: 0737 68655
Telex: 947234
CH,CS,E,P

SALES & SUPPORT OFFICES

Arranged alphabetically by country

GREAT BRITAIN (Cont'd)

Hewlett-Packard Ltd.
Avon House
435 Stratford Road
Shirley, **SOLIHULL**, West Midlands
B90 4BL
Tel: 021 745 8800
Telex: 339105
CH,CS,E,P

Hewlett-Packard Ltd.
West End House
41 High Street, West End
SOUTHAMPTON
Hampshire SO3 3DQ
Tel: 04218 6767
Telex: 477138
CH,CS,P

Hewlett-Packard Ltd.
Eskdale Rd.
Winnersh, **WOKINGHAM**
Berkshire RG11 5DZ
Tel: 0734 696622
Telex: 848884
E

Hewlett-Packard Ltd.
King Street Lane
Winnersh, **WOKINGHAM**
Berkshire RG11 5AR
Tel: 0734 784774
Telex: 847178
A,CH,CS,E,M,MP,P

Hewlett-Packard Ltd.
Nine Mile Ride
Easthampstead, **WOKINGHAM**
Berkshire, 3RG11 3LL
Tel: 0344 773100
Telex: 848805
CH,CS,E,P

IRELAND

NORTHERN IRELAND

Hewlett-Packard Ltd.
Cardiac Services Building
95A Finaghy Road South
BELFAST BT10 0BY
Tel: 0232 625-566
Telex: 747626
CH,CS

SCOTLAND

Hewlett-Packard Ltd.
SOUTH QUEENSFERRY
West Lothian, EH30 9TG
Tel: 031 331 1188
Telex: 72682
CH,CM,CS,E,M,P

UNITED STATES

Alabama

Hewlett-Packard Co.
700 Century Park South, Suite 128
BIRMINGHAM, AL 35226
Tel: (205) 822-6802
A,CH,M

Hewlett-Packard Co.
420 Wynn Drive
HUNTSVILLE, AL 35805
P.O. Box 7700
HUNTSVILLE, AL 35807
Tel: (205) 830-2000
CH,CM,CS,E,M*

Arizona

Hewlett-Packard Co.
8080 Pointe Parkway West
PHOENIX, AZ 85044
Tel: (602) 273-8000
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
2424 East Aragon Road
TUCSON, AZ 85706
Tel: (602) 889-4631
CH,E,MS**

California

Hewlett-Packard Co.
99 South Hill Dr.
BRISBANE, CA 94005
Tel: (415) 330-2500
CH,CS

Hewlett-Packard Co.
P.O. Box 7830 (93747)
5060 E. Clinton Avenue, Suite 102
FRESNO, CA 93727
Tel: (209) 252-9652
CH,CS,MS

Hewlett-Packard Co.
P.O. Box 4230
1430 East Orangethorpe
FULLERTON, CA 92631
Tel: (714) 870-1000
CH,CM,CS,E,MP

Hewlett-Packard Co.
320 S. Kellogg, Suite B
GOLETA, CA 93117
Tel: (805) 967-3405
CH

Hewlett-Packard Co.
5400 W. Rosecrans Boulevard
LAWNDALE, CA 90260
P.O. Box 92105
LOS ANGELES, CA 90009
Tel: (213) 970-7500
Telex: 910-325-6608
CH,CM,CS,MP

Hewlett-Packard Co.
3155 Porter Oaks Drive
PALO ALTO, CA 94304
Tel: (415) 857-8000
CH,CS,E

Hewlett-Packard Co.
4244 So. Market Court, Suite A
P.O. Box 15976
SACRAMENTO, CA 95852
Tel: (916) 929-7222
A*,CH,CS,E,MS

Hewlett-Packard Co.
9606 Aero Drive
P.O. Box 23333
SAN DIEGO, CA 92139
Tel: (619) 279-3200
CH,CM,CS,E,MP

Hewlett-Packard Co.
2305 Camino Ramon "C"
SAN RAMON, CA 94583
Tel: (415) 838-5900
CH,CS

Hewlett-Packard Co.
3005 Scott Boulevard
SANTA CLARA, CA 95050
Tel: (408) 988-7000
Telex: 910-338-0586
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
5703 Corsa Avenue
WESTLAKE VILLAGE, CA 91362
Tel: (213) 706-6800
E*,CH*,CS*

Colorado

Hewlett-Packard Co.
24 Inverness Place, East
ENGLEWOOD, CO 80112
Tel: (303) 649-5000
A,CH,CM,CS,E,MS

Connecticut

Hewlett-Packard Co.
47 Barnes Industrial Road South
P.O. Box 5007
WALLINGFORD, CT 06492
Tel: (203) 265-7801
A,CH,CM,CS,E,MS

Florida

Hewlett-Packard Co.
2901 N.W. 62nd Street
P.O. Box 24210
FORT LAUDERDALE, FL 33307
Tel: (305) 973-2600
CH,CS,E,MP

Hewlett-Packard Co.
6177 Lake Ellenor Drive
P.O. Box 13910
ORLANDO, FL 32859
Tel: (305) 859-2900
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
5750B N. Hoover Blvd., Suite 123
P.O. Box 15200
TAMPA, FL 33614
Tel: (813) 884-3282
A*,CH,CM,CS,E*,M*

Georgia

Hewlett-Packard Co.
2000 South Park Place
P.O. Box 105005
ATLANTA, GA 30348
Tel: (404) 955-1500
Telex: 810-766-4890
A,CH,CM,CS,E,MP

Hawaii

Hewlett-Packard Co.
Kawaiahao Plaza, Suite 190
567 South King Street
HONOLULU, HI 96813
Tel: (808) 526-1555
A,CH,E,MS

Illinois

Hewlett-Packard Co.
304 Eldorado Road
P.O. Box 1607
BLOOMINGTON, IL 61701
Tel: (309) 662-9411
CH,MS**

Hewlett-Packard Co.
1100 31st Street, Suite 100
DOWNERS GROVE, IL 60515
Tel: (312) 960-5760
CH,CS

Hewlett-Packard Co.
5201 Tollview Drive
ROLLING MEADOWS, IL 60008
Tel: (312) 255-9800
Telex: 910-687-1066
A,CH,CM,CS,E,MP

Indiana

Hewlett-Packard Co.
7301 No. Shadeland Avenue
P.O. Box 50807
INDIANAPOLIS, IN 46250
Tel: (317) 842-1000
A,CH,CM,CS,E,MS

Iowa

Hewlett-Packard Co.
1776 22nd Street, Suite 1
WEST DES MOINES, IA 50265
Tel: (515) 224-1435
CH,MS**

Kansas

Hewlett-Packard Co.
7804 East Funston Road, #203
WICHITA, KS 67207
Tel: (316) 684-8491
CH

Kentucky

Hewlett-Packard Co.
10300 Linn Station Road, #100
LOUISVILLE, KY 40223
Tel: (502) 426-0100
A,CH,CS,MS

Louisiana

Hewlett-Packard Co.
160 James Drive East
ST. ROSE, LA 70087
P.O. Box 1449
KENNER, LA 70063
Tel: (504) 467-4100
A,CH,CS,E,MS

Maryland

Hewlett-Packard Co.
3701 Koppers Street
BALTIMORE, MD 21227
Tel: (301) 644-5800
Telex: 710-862-1943
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
2 Choke Cherry Road
ROCKVILLE, MD 20850
Tel: (301) 948-6370
A,CH,CM,CS,E,MP

Massachusetts

Hewlett-Packard Co.
1775 Minuteman Road
ANDOVER, MA 01810
Tel: (617) 682-1500
A,C,CH,CS,CM,E,MP,P*

Hewlett-Packard Co.
32 Hartwell Avenue
LEXINGTON, MA 02173
Tel: (617) 861-8960
CH,CS,E

Michigan

Hewlett-Packard Co.
4326 Cascade Road S.E.
GRAND RAPIDS, MI 49506
Tel: (616) 957-1970
CH,CS,MS

Hewlett-Packard Co.
1771 W. Big Beaver Road
TROY, MI 48084
Tel: (313) 643-6474
CH,CS

Minnesota

Hewlett-Packard Co.
2025 W. Larpenteur Ave.
ST. PAUL, MN 55113
Tel: (612) 644-1100
A,CH,CM,CS,E,MP

Missouri

Hewlett-Packard Co.
11131 Colorado Avenue
KANSAS CITY, MO 64137
Tel: (816) 763-8000
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
13001 Hollenberg Drive
BRIDGETON, MO 63044
Tel: (314) 344-5100
A,CH,CS,E,MP

SALES & SUPPORT OFFICES

Arranged alphabetically by country

UNITED STATES (Cont'd)

Nebraska
Hewlett-Packard
10824 Old Mill Rd., Suite 3
OMAHA, NE 68154
Tel: (402) 334-1813
CM,MS

New Jersey
Hewlett-Packard Co.
120 W. Century Road
PARAMUS, NJ 07652
Tel: (201) 265-5000
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
60 New England Av. West
PISCATAWAY, NJ 08854
Tel: (201) 981-1199
A,CH,CM,CS,E

New Mexico
Hewlett-Packard Co.
11300 Lomas Blvd.,N.E.
P.O. Box 11634
ALBUQUERQUE, NM 87112
Tel: (505) 292-1330
CH,CS,E,MS

New York
Hewlett-Packard Co.
5 Computer Drive South
ALBANY, NY 12205
Tel: (518) 458-1550
A,CH,E,MS

Hewlett-Packard Co.
9600 Main Street
P.O. Box AC
CLARENCE, NY 14031
Tel: (716) 759-8621
CH

Hewlett-Packard Co.
200 Cross Keys Office Park
FAIRPORT, NY 14450
Tel: (716) 223-9950
CH,CM,CS,E,MS

Hewlett-Packard Co.
7641 Henry Clay Blvd.
LIVERPOOL, NY 13088
Tel: (315) 451-1820
A,CH,CM,E,MS

Hewlett-Packard Co.
No. 1 Pennsylvania Plaza
55th Floor
34th Street & 8th Avenue
MANHATTAN NY 10119
Tel: (212) 971-0800
CH,CS,E*,M*

Hewlett-Packard Co.
250 Westchester Avenue
WHITE PLAINS, NY 10604
Tel: (914) 684-6100
CM,CH,CS,E

Hewlett-Packard Co.
3 Crossways Park West
WOODBURY, NY 11797
Tel: (516) 921-0300
A,CH,CM,CS,E,MS

North Carolina
Hewlett-Packard Co.
5605 Roanne Way
P.O. Box 26500
GREENSBORO, NC 27420
Tel: (919) 852-1800
A,CH,CM,CS,E,MS

Ohio
Hewlett-Packard Co.
9920 Carver Road
CINCINNATI, OH 45242
Tel: (513) 891-9870
CH,CS,MS

Hewlett-Packard Co.
16500 Sprague Road
CLEVELAND, OH 44130
Tel: (216) 243-7300
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
962 Crupper Ave.
COLUMBUS, OH 43229
Tel: (614) 436-1041
Eff: Nov. 25, 1983
675 Brooksedge Blvd.
WESTERVILLE, OH 43081
CH,CM,CS,E*

Hewlett-Packard Co.
330 Progress Rd.
DAYTON, OH 45449
Tel: (513) 859-8202
A,CH,CM,E*,MS

Oklahoma
Hewlett-Packard Co.
304 N. Meridian, Suite A
P.O. Box 75609
OKLAHOMA CITY, OK 73147
Tel: (405) 946-9499
A*,CH,E*,MS

Hewlett-Packard Co.
3840 S. 103rd E. Avenue, #100
P.O. Box 35747
TULSA, OK 74153
Tel: (918) 665-3300
A**,CH,CS,M*

Oregon
Hewlett-Packard Co.
9255 S. W. Pioneer Court
P.O. Box 328
WILSONVILLE, OR 97070
Tel: (503) 682-8000
A,CH,CS,E*,MS

Pennsylvania
Hewlett-Packard Co.
111 Zeta Drive
PITTSBURGH, PA 15238
Tel: (412) 782-0400
A,CH,CS,E,MP

Hewlett-Packard Co.
2750 Monroe Boulevard
P.O. Box 713
VALLEY FORGE, PA 19482
Tel: (215) 666-9000
A,CH,CM,E,M

South Carolina
Hewlett-Packard Co.
Brookside Park, Suite 122
1 Harbison Way
P.O. Box 21708
COLUMBIA, SC 29221
Tel: (803) 732-0400
CH,E,MS

Hewlett-Packard Co.
Koger Executive Center
Chesterfield Bldg., Suite 124
GREENVILLE, SC 29615
Tel: (803) 297-4120

Tennessee
Hewlett-Packard Co.
224 Peters Road, Suite 102
P.O. Box 22490
KNOXVILLE, TN 37922
Tel: (615) 691-2371
A*,CH,MS

Hewlett-Packard Co.
3070 Directors Row
MEMPHIS, TN 38131
Tel: (901) 346-8370
A,CH,MS

Texas
Hewlett-Packard Co.
4171 North Mesa
Suite C-110
EL PASO, TX 79902
Tel: (915) 533-3555
CH,E*,MS**

Hewlett-Packard Co.
10535 Harwin Drive
P.O. Box 42816
HOUSTON, TX 77042
Tel: (713) 776-6400
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
930 E. Campbell Rd.
P.O. Box 1270
RICHARDSON, TX 75080
Tel: (214) 231-6101
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
1020 Central Parkway South
P.O. Box 32993
SAN ANTONIO, TX 78216
Tel: (512) 494-9336
CH,CS,E,MS

Utah
Hewlett-Packard Co.
3530 W. 2100 South
SALT LAKE CITY, UT 84119
Tel: (801) 974-1700
A,CH,CS,E,MS

Virginia
Hewlett-Packard Co.
4305 Cox Road
GLEN ALLEN, VA 23060
P.O. Box 9669
RICHMOND, VA 23228
Tel: (804) 747-7750
A,CH,CS,E,MS

Washington
Hewlett-Packard Co.
15815 S.E. 37th Street
BELLEVUE, WA 98006
Tel: (206) 643-4000
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
Suite A
708 North Argonne Road
SPOKANE, WA 99212
Tel: (509) 922-7000
CH,CS

West Virginia
Hewlett-Packard Co.
4604 MacCorkle Ave.
P.O. Box 4297
CHARLESTON, WV 25304
Tel: (304) 925-0492
A,MS

Wisconsin
Hewlett-Packard Co.
150 S. Sunny Slope Road
BROOKFIELD, WI 53005
Tel: (414) 784-8800
A,CH,CS,E*,MP

URUGUAY

Pablo Ferrando S.A.C. e I.
Avenida Italia 2877
Casilla de Correo 370
MONTEVIDEO
Tel: 80-2586
Telex: Public Booth 901
A,CM,E,M

VENEZUELA

Hewlett-Packard de Venezuela C.A.
3RA Transversal Los Ruices Norte
Edificio Segre 1, 2 & 3
Apartado 50933
CARACAS 1071
Tel: 239-4133
Telex: 251046 HEWPACK
A,CH,CS,E,MS,P

Hewlett-Packard de Venezuela C.A.
Calle-72-Entre 3H y 3Y, No. 3H-40
Edificio Ada-Evelyn, Local B
Apartado 2646
4001, **MARACAIBO**, Estado Zulia
Tel: (061) 80.304
C,E*

Hewlett-Packard de Venezuela C.A.
Calle Vargas Rondon
Edificio Seguros Carabobo, Piso 10
VALENCIA
Tel: (041) 51 385
CH,CS,P

Bioelectronica Medica C.A.
Calle Buen Pastor
Edif. Cota Mil-Piso 2 y Semi Sotano 1
Boleita Norte
Apartado 50710 ***CARACAS*** *1050A*
Tel: 239 84 41
Telex: 26518

ZIMBABWE

Field Technical Sales
45 Kelvin Road, North
P.B. 3458
SALISBURY
Tel: 705 231
Telex: 4-122 RH
C,E,M,P

July 1983 5952-6900

Indicates main office

HP distributors are printed in italics.