

# Chapter 3

## The Workabout Command Processor

### The Workabout Command Processor

---

#### Overview

The Workabout is provided, (for development purposes), with two complementary system interfaces.

- The Command Processor which is similar, though significantly different, to the Command Processor on the Psion HC range of machines. See *Appendix B* of this manual for details of these differences.
- The System Screen which is similar to the System Screen of the Series 3a machine. See *Appendix C* of this manual for details of these differences. See the chapter *The System Screen* in the *Series 3 Programmers Reference* manual and the *Series 3/3a User Guide* for a full description.

The choice of System Interface "System:interface" is made by pressing the Menu key, whilst in the Startup Shell "Startup:Shell", to bring up the Special menu. Either "System Screen" or "Command processor" is then selected. This can be done either when the Workabout is started for the first time, or after exiting all other processes on the machine. Note that it is not possible to get this Special menu if an *autoexec* program is present on the machine, since this will execute automatically instead.

The Workabout Command Processor provides an MS-DOS like utility for functions that can be executed from a command line. In addition to the familiar DOS commands there are several commands peculiar to the Workabout. Some provide information, such as *LSEG*. Others invoke features of the Workabout which are beyond the scope of DOS, such as *START* which runs processes asynchronously. The range of functionality covered includes file and SSD management, program management, information requests, and Workabout configuration. See below for the differences from MS-DOS.

Commands can be entered one at a time by typing at the Command Processor in response to a > prompt. Alternatively *batch files* can be invoked, to run a series of commands consecutively - or simply to cut down on typing. For example, typing *xbat* is shorthand for invoking all the commands stored in the file *xbat.btf*. Note that batch files must have the extension *.btf*.

Differences between the Workabout Command Processor and the MS-DOS command line or Psion HC Command Processor are described in *Appendix B - Differences between the HC Command Processor, the Workabout Command Processor and MS-DOS*.

## Command Processor menusXE "Command Processor:menus"§tc "Command Processor menus" ¶ 3§

If the Menu key is pressed at any time when using the *Workabout* Command Processor, then a menu bar is presented.

**All** options may be accessed using "hotkey" key combinations, ("accelerators"). These are shown in the menus adjacent to the relevant option, and are given in square brackets, ( [ ] ), in the table below.

For the default machine settings see the SETDEF command in the Alphabetical listing **section below**.

Menu name	Option	Function
Time	Time and date [Psion-TXE "Hotkey:Psion-T"\$XE "Psion-T hotkey"\$]	Set the system time and dateXE "Set:system time and date"\$XE "Date:set"\$XE "Time:set"\$, (see Formats)
	Summer timeXE "Time:summer time"\$ [Shift-Psion-SXE "Hotkey:Shift-Psion-S"\$XE "Shift-Psion-S hotkey"\$]	Set summer time On or OffXE "Set:summer time On or Off"\$XE "Summer time, set On or Off"\$
	FormatsXE "Time:formats"\$ [Psion-FXE "Hotkey:Psion-F"\$XE "Psion-F hotkey"\$]	Set dateXE "Date:format, set"\$XE "Set:date format"\$ to be displayed in one of three formats: Day month year Year month day Month day year  Set the date separatorXE "Separator:set for date"\$XE "Date:separator, set"\$XE "Set:date separator"\$ to any non-alphanumeric character.

Set the time formatXE "Set:time  
format"\$XE "Time:format, set"\$ to  
either am-pm or 24 hour.

Set the time separatorXE  
"Separator:set for time"\$XE  
"Time:separator, set"\$XE "Set:time  
separator"\$ to any non-alphanumeric  
character.

Set the "start of week" dayXE "Start of  
week day, set"\$XE "Set:\"start of  
week\" day"\$ to any day of the week  
from Monday to Sunday. The default is  
Monday.

Control	SoundXE "Control:Sound"\$XE "Sound"\$ [Psion-SXE "Hotkey:Psion-S"\$XE "Psion-S hotkey"\$]	Set all sounds to be On or OffXE "Sounds, set all On/Off"\$XE "Set:all sounds On/Off"\$.  Set beeps to be Loud, Quiet or OffXE "Beeps, set"\$XE "Set:beeps"\$.  Set key clicksXE "Set:key clicks"\$ to be Loud, Quiet or Off.
	Auto switch offXE "Control:Auto switch off"\$XE "Auto switch off"\$ [Psion-AXE "Hotkey:Psion-A"\$XE "Psion-A hotkey"\$]	Set auto switch offXE "Set:auto switch off"\$ of the machine to Yes or No or "If no external power".  Set the auto switch off timeXE "Set:auto switch off time"\$ of the machine to any whole number of minutes between one and thirty.  Set auto switch off of the backlightXE "Set:backlight auto switch off"\$ to Yes or No.  Set the auto switch off timeXE "Set:backlight auto switch off time"\$ of the backlight to any whole number of minutes between one and ten.  Set the backlight keyXE "Set:backlight key"\$ as Disabled or Enabled.
	Special keyboardXE "Control:Special keyboard"\$XE "Special	Switch between "Special keyboardXE "Set:keyboard"\$ selected" and "Standard keyboard selected". There is no dialog.

<pre>keyboard"\$ [Psion-KXE "Hotkey:Psio n-K"\$XE "Psion-K hotk ey"\$]</pre>
--

Special	Remote linkXE "Special:Remote link"\$XE "Remote link"\$ [Psion-LXE "Hotkey:Psion-L"\$XE "Psion-L hotkey"\$]	Set remote linkXE "Set:remote link"\$ On or Off  Set the baud rateXE "Set:baud rate"\$ to one of 300, 600,1200, 2400, 4800, 9600 or 19200.  Set the communications portXE "Set:communications port"\$ to A, B or C.  Set any extra parametersXE "Set:extra comms parameters"\$ required.
	Wrap onXE "Special:Wrap on"\$XE "Wrap on/off"\$ [Psion-WXE "Hotkey:Psion-W"\$XE "Psion-W hotkey"\$]	Switch between "Wrap is on" and "Wrap is off". When WrapXE "Set:wrap"\$ is <b>on</b> , displayed text does not disappear off the edge of the screen, but continues on the next line. There is no dialog.
	Zoom inXE "Special:Zoom in"\$XE "Zoom:in"\$ [Psion-ZXE "Hotkey:Psion-Z"\$XE "Psion-Z hotkey"\$]	Increase the character font sizeXE "Character font size:increase"\$XE "Font:size, increase"\$ .

	<div>Zoom outXE "Special:Zoom out"\$XE "Zoom:out"\$ [Shift-Psion-ZXE "Hotkey:Shift -Psion-Z"\$XE "Shift-Psion-Z hotk ey"\$]</div>	<div>Decrease the character font size.XE "Character font size:decrease"\$XE "Font:size, decrease"\$</div>
	<div>ExitXE "Special:Exit" \$XE "Exit"\$ [Psion-XXE "Psion-X hotk ey"\$XE "Hotkey:Psio n-X"\$]</div>	<div>Return to the Startup Shell (Psion logo), or other running process (if any)</div>



**Notes:**

When several options for a setting are available, a left pointing and right pointing arrow will appear bracketing the current setting.

Selection is then made using the left and right arrow keys.

Alternatively, pressing Tab will display the full list of options, which can be selected using the arrow keys followed by Enter. If there are a lot of options in a list an arrow pointing up or down will be shown in the corner of the list window. The other options can then be revealed by using the up or down arrow key as required. **When a numeric entry is required, enter digits or press > (Psion+Full-stop) to cycle upwards, or press < (Psion-0) to cycle downwards.**

For full details of the differences between the Psion Series 3a menus and the Workabout menus see *Appendix C - Differences between the Series 3a System Screen and the Workabout System Screen and Command Processor Menus.*

## Font sizesXE "Font:size"\$ and zoom settingsXE "Zoom:settings"\$tc "Font sizes and zoom settings" \l 3§

The *Workabout* can display characters in **many fonts**. The **Command processor uses** five different font sizes. The use of "zoom in " and "zoom out" increases or decreases font size, (see above). Zooming in (from a larger to a smaller font) will re-display some of the information that scrolls off the top of the screen in a larger font size.

The five different zoom settings give the following number of characters per line and lines of text on the screen.

Zoom setting	1	2	3	4	5
Characters per line	39	29	27	23	18
Number of lines	12	9	7	6	5
Font size	6	8	11	13	16

## Batch file processingXE "Batch file:processing"\$tc "Batch file processing" \l 3§

Batch files are plain text files, with each separate command being on a line on its own. By default, batch files have extension *.btf*. They are normally expected to be stored in the *\btf* directory. The reason that the extension *.bat* is not used is to prevent accidental execution of *Workabout* batch files on a PC (with potentially disastrous consequences), or accidental execution of PC batch files on the *Workabout*

To invoke a batch file with name *backup.btf*, just type *backup* at the > prompt in the

Command Processor. This will launch the *backup.btf* batch file, unless another executable file exists of the same name but with a different extension, (see *Launching programs* below).

If necessary, the full path of the batch file should be given too. Thus:

```
loc::b:\batch\backup
```

or

```
rem::c:\workabt\devp\restore.btf
```

with the extension needing to be supplied in case it differs from *.btf*. **Note that loc:: is the local filing system and rem:: is the remote filing system.**

Batch files are always run *synchronously*. This means that while the batch file is executing, the user **cannot** task back to the Command Processor and continue to issue other commands. No additional commands can be typed into the Command Processor until any batch files it is executing have completed. If the batch file name is preceded by *start*, (see *Launching programs* below), it will still be run synchronously, thus:

```
start loc::b:\batch\backup
```

is exactly the same as:

```
loc::b:\batch\backup
```

Batch files can also call other batch files.

If more complex functionality is required, a *program* should be written, to replace the batch file. This program could be written either in OPL, or in another high level language such as C.

Tip: Any text on a line preceded by *REM* (and a following space) becomes a comment that is not executed when the batch file is run. This is useful for temporarily removing a command line, for example when debugging a batch file. Another way is to precede it with a colon (:) at the start of the line. This converts the command into a label.

## Launching programsXE "Programs:launching"\$XE "Launching programs"\$tc "Launching programs" \l 3\$

As well as running batch files, the Command Processor can be used to launch user developed programs - either OPL programs (with extension *.opo*), OPL applications (with extension *.opa*) or EPOC executables (typically with extension *.img* or *.app*).

Like batch files, these other programs can be run simply by typing their name, or by preceding the name by *START* (see below for synchronous/asynchronous launching).

On the Workabout there is a fixed default directory structure, which is searched automatically for program files if full details are not provided of path and filename extension.

The Command Processor checks that there is no internal command with the name given. The Command Processor then looks for filenames in the following order of extensions, (if no extension is given):

1. *.img*
2. *.app*
3. *.opo*

4. *.opa*
5. *.btf*

The order of searching on the *Workabout* is illustrated by the following examples:

### No path or extension

Enter:

xxx

1. The Command Processor will look for *xxx.\** in the current directory (where *\** is one of the above five extensions). If found, it runs it.

So far the same rules as in DOS have applied. Next "Psion Rules" are followed. In each case below all drives A:, B:, C: (if present), and M:, (signified by *drive:* ), are searched in that order. The Command Processor looks for:

2. a *.img* file in *drive:\img\*
3. a *.app* file in *drive:\app*
4. a *.opo* file in *drive:\opo\*
5. a *.opa* file in *drive:\app\*
6. and finally a *.btf* file in *drive:\btf\*

### An extension is given

Enter:

XXX.APP

The procedure will be the same as above except that the given file *xxx.app* is looked for only in the corresponding directory on all the drives, in this case *\app*.

### A particular drive or directory is given

Enter, for example:

M:XXX

or

SUBDIR\XXX

DOS rules only apply. The specified file *xxx* (no extension) will be looked for in the specified path. You can specify a valid extension, if you like, and it will be respected. [See \*Files and directories\* below.](#)

### Notes

As can be seen, the file *program.img* will be found in preference to a file *program.opo* if no extension is given. To ensure that the file *program.opo* is run, enter the extension explicitly:

program.opo

Additional parameters can be passed to programs. For example,

dojob b:

## **Synchronous programs**`XE "Programs:synchronous"$XE "Synchronous programs"$ and asynchronous programs``XE "Programs:asynchronous"$XE "Asynchronous programs"$tc "Synchronous programs and asynchronous programs" \l 3$`

In contrast to batch files, which are always run synchronously (see above), programs can be run either synchronously or, exploiting the multi-tasking capabilities of the *Workabout*, asynchronously.

By default, programs are launched synchronously, if their name is entered on its own. This means that while the program is executing, the user **cannot** task back to the Command Processor and continue to issue other commands.

To run a program asynchronously, prefix the program name with the keyword `START`. This means that while the program is executing, the user can task back to the Command Processor and continue to issue other commands.

When the program is started, it will by default (assuming that it has some user interface) take over the foreground screen. To access the Command Processor, or indeed any other tasks that may be running on the *Workabout* at the time, press **Psion-Tab**, ("Switch task"), as many times as is required. Every time **Psion-Tab** is pressed, a different program cycles into foreground.

Note that there is no need to quit the foreground program in order to start another. **Start** a new program by pressing the **Psion-Tab** key combination until you get into the Command Processor, then type the name of the program at the command line, (with or without the preceding `START`, as required).

However, users should avoid starting up new programs unnecessarily - since each additional program reduces the memory available for the programs already running, **and uses up other resources and processor time**.

## **Terminating programs**`XE "Programs:terminating"$XE "Terminate:a program"$tc "Terminating programs" \l 3$`

Many programs contain mechanisms within themselves for the user to terminate them. For example, they may contain an `Exit` command.

Alternatively, programs can be terminated from the Command Processor, using either the `STOP` or `KILL` commands. As is explained in the alphabetical listing below, `STOP` should be used in preference to `KILL` whenever possible. The relevant menu options should be used to stop the built-in Psion applications.

Note that if a program is run synchronously, it is not possible to terminate it by tasking to the Command Processor that launched it - since that Command Processor is inaccessible until the program terminates.

There is also a "super kill" key combination, **Psion-Shift-Ctrl-K**, which kills the current foreground process. In extreme circumstances, recourse to resetting the *Workabout* may be required.

## The command line editor

### "Editor:command line"

### "Command line editor"

### 3

Up to 32 previous commands can be reviewed by means of the Up and Down cursor keys at the command line. Any previous command displayed in this way can be edited before being issued again.

To clear the command line at any time, press Esc.

As would be expected, each individual command is entered to the *Workabout* by pressing Enter after typing its name. The command name can **never** be abbreviated. **Built-in commands** are not case sensitive **and neither are their parameters/options**. Parameters to user-developed applications **may** be case sensitive, however.

Certain other keys and key combinations also have special effects in the command line editor, (or when a batch file is running).

xe Terminates a line of input

"Terminate:line of input"

"Line

"Key:Enter" Enter of input, terminate

xe "Key:Left" Move the cursor left or right

"Cursor:move left"

"Key:Right" Left/ "Cursor:move right" along the line being edited.

Right

xe "Key:Up" Recall previous commands

"Command:recall previous"

"Key:Down" Up/ "Recall previous commands". The up and down arrow keys move

Down backwards and forwards respectively through the last 32

commands (maximum) that were issued.

xe "Key:Del" Del on its own deletes the character **preceding the cursor**, or any

highlighted text (see below).

"Delete:preceding character"

Esc "Key:Esc" Esc on its own deletes the entire line.

"Delete:entire line"

Also: break out of a batch file or a long listing, or cancel a `WAIT` command

There are also several special keypress combinations:

xe "Key Deletes the character following the cursor, (if no text is  
combination:Shift- highlighted).XE "Delete:following character"\$  
Del"\$Shift-Del

xe "Key Deletes from the cursor to the start of the line, (if no text is  
combination:Psion-highlighted).XE "Delete:to start of line"\$  
Del"\$Psion-Del

xe "Key Deletes from the cursor to the end of the lineXE "Delete:to end of  
combination:Shift- line"\$, (if no text is highlighted).  
Psion-Del"\$Shift-  
Psion-Del

xe "Key Moves the cursor left, highlighting (in reverse video) all characters  
combination:Shift- passed. If the Shift key is kept depressed and Right is then pressed  
Left"\$Shift-Left the text will be unhighlighted again as the cursor passes.XE  
"Highlight:characters passed to the left"\$

xe "Key Moves the cursor right, highlighting (in reverse video) all characters  
combination:Shift- passed. If the Shift key is kept depressed and Left is then pressed  
Right"\$Shift-Right the text will be unhighlighted again as the cursor passes.XE  
"Highlight:characters passed to the right"\$

xe "Key Highlights (in reverse video) the current word.XE  
combination:Psion-"Highlight:current word"\$  
Shift-Ctrl-Left"\$Ps  
ion-Shift-Ctrl-Left

xe "Key Highlights (in reverse video) the current lineXE "Highlight:current  
combination:Psion-line"\$.  
Shift-Ctrl-Right"\$PThis is somewhat redundant **in this instance**, since all you can then  
sion-Shift-Ctrl-Rig do is delete the line (which is more easily done by simply pressing  
ht Esc). **However, the same editor is available for use by user-  
developed applications.**

xe "Key                      Moves the cursor to the beginning of the current wordXE  
 combination:Ctrl-L "Cursor:to beginning of current word"\$ being edited, or of the  
 eft"\$Ctrl-Left              preceding word if already in that position.

xe "Key                      Moves the cursor to the beginning of the next word to the rightXE  
 combination:Ctrl-R "Cursor:to beginning of next word right"\$, or to the end of the line  
 ight"\$Ctrl-Right        if already on the last word.

xe "Key                      Moves the cursor to the beginning of the lineXE "Cursor:to  
 combination:Psion-beginning of line"\$ being edited.  
 Left"\$Psion-Left

xe "Key                      Moves the cursor to the end of the lineXE "Cursor:to end of line"\$  
 combination:Psion-being edited.  
 Right"\$Psion-Righ  
 t

xe "Key                      Recalls last command issuedXE "Command:last issued"\$XE "Last  
 combination:Psion-command issued"\$.  
 Down"\$Psion-Dow  
 n

xe "Key                      Recalls earliest recallable commandXE "Earliest recallable  
 combination:Psion-command"\$XE "Command:earliest recallable"\$.  
 Up"\$Psion-Up

xe "Key                      The "Help" hotkeyXE "Help:hotkey"\$ combination. It brings up the  
 combination:Shift- Command Processor Help menu.  
 Esc"\$Shift-Esc



xe "Key The "Help index" hotkeyXE "Help:index hotkey"\$ combination. It  
 combination:Ctrl-S is used to bring up a Help index.  
 hift-Esc"\$Ctrl-Shift-  
 -Esc

xe "Key The "Task" hotkeyXE "Task hotkey"\$ combination. It allows  
 combination:Psion-switching between the Command Processor and other tasks, (if  
 Tab"\$Psion-Tab any).

xe "Key The "Break" hotkeyXE "Break hotkey"\$ combination cancels a WAIT  
 combination:Ctrl- command. **This has the same effect as Esc.**  
 C"\$Ctrl-C

xe "Key The "Pause" hotkeyXE "Pause:hotkey"\$ combination. Pauses a  
 combination:Ctrl- batch file or a long listing.  
 S"\$Ctrl-S

## PausingXE "Pause:the screen display"\$ the screen displaytc "Pausing the screen display" \l 3§

All commands automatically pause when a screenful of information has been displayed. In all cases, pressing Enter resumes the display. The Esc key can usually be used to terminate the command listing.

Commands do not pause when being executed within a batch file.

## Running multiple System Interfacestc "Running multiple System Interfaces" \l 3§

It is not possible to run an additional copy of the Command Processor. If this is attempted by entering at the command line prompt of the Command Processor:

```
start sys$cmdp
```

a "File already exists" dialog will appear.

It is, however, possible to run the System Screen at the same time as the Command Processor. To do this enter at the command line prompt of the Command Processor:

```
start sys$gsys
```

Only one copy of the System Screen may be started - if one is already running a "File

already exists" dialog will appear.

The Task key combination (Psion-Tab) can then be used to switch between the system interfaces, (and any other tasks that are also running).

It is neither possible to install the Command Processor in the System Screen, nor start the Command Processor from the System Screen.

## **Exiting the Command Processor** **Command Processor:exiting"\$XE "Exit:the Command Processor"\$tc** **"Exiting the Command Processor" \l 3§**

The command processor can be exited (closed down) in three ways:

- enter EXIT
- select the 'Exit' option from the 'Special' menu
- press Psion-X

If no other process **is** running, you will return to the Startup Shell.

---

## **Files and directories** **\$tc "Files and directories" \l 2§**

### **File In Use error messages** **\$XE "Error message, 'File In Use'"\$XE** **"File In Use:error message"\$tc "File In Use error messages" \l 3§**

If any *Workabout* application has a file open for editing purposes, no other application is allowed to alter that file in any way.

For this reason, some file commands may unexpectedly fail, when issued from the Command Processor.

For example, it is impossible to make a complete copy of all the files in a directory if any one (or more) of these files is in use by another application. Commands to delete or rename such a file will likewise report an error.

### **Default path** **\$XE "Default:path"\$XE "Path, default"\$ and current** **directory** **\$XE "Current:directory"\$XE "Directory:current"\$tc** **"Default path and current directory" \l 3§**

Unlike MS-DOS, in which there is always a current logged directory *for each different drive* (and so there are as many current directories as there are drives), in EPOC there is only one current directory for each running process. Indeed, instead of referring to a *current directory*, it is preferable to refer to the *current path*, since the drive is included too (and the filing system).

Unlike on the Psion HC, there is no single command to set the default path on the *Workabout*. The drive letter followed by a colon should be used to set the current drive in the Command Processor or in batch files , and the `cd` command to change the current directory.

Note that if the current path is *a:\work\*, the effect of typing `cd b:\play` is **the same** in the Workabout Command Processor as in MS-DOS, but is limited to local drives.

However, in EPOC, there is a different current path for each current application. Correctly written applications (*.app* and *.opa* programs) tell EPOC their current path. Changing the path in one application does not alter the path in another application. This may be regarded as a significant improvement over MS-DOS, where the currently logged directories in the command processor are often annoyingly altered by running (synchronously!) another process. By the time the second process has terminated, the MS-DOS command processor may have been logged to a different drive and a different directory.

Note that any changes to the path of the Command Processor inside a batch file will continue to have effect after the termination of the batch file, since no new process is run up just by virtue of a batch file being executed.

### **Specifying file names** **File:names"\$ as command parameters** **Command:parameters"\$** **Parameters, command"\$** **Specifying file names as command parameters" \l 3\$**

In all, fully specified filenames are regarded as having *five* parts:

1. A *filing system* (e.g. *loc::* or *rem::*).
2. A *drive* or *device* (e.g. *b:*).
3. A *path* (e.g. *\* or *\accounts\jan\*).
4. A *filename* (e.g. *job*).
5. An *extension* (e.g. *.img*).

Frequently, it is possible to omit part of the full specification of the path of a file, when giving a filename as a parameter to a command in the Command Processor. Any missing parts of the filename will be filled in from the current path. In the *Alphabetical listing* below *filespec* implies a full file specification as listed above, *filename* means the name of the file only.

For example, if the current path is *m:\img\*, the command `attrib job.img -r` operates on the file with full specification *m:\img\job.img*. However, the command `att b:\backup\job.img` operates, naturally enough, on the file *b:\backup\job.img* - regardless of the current path.

If the current directory on *b:* is *\img\*, the command `attrib b:job.img` operates on a file *b:\img\job.img* (if one exists) - not on any file *job.img* in the root directory of *b:*. The reason for this is that the filename *b:job.img* is interpreted as having only the three parts:

- a drive (*b:*)
- a basic name (*job*)
- an extension (*.img*).

However, the *path* component is missing, and so this is taken from the current path for the drive.

To specify a file *job.img* on the root directory of *b:*, type:

```
b:\job.img
```

including the crucial \ character.

## Specifying paths as command parameterstc "Specifying paths as command parameters" ¶ 3§

Just as it is often possible to omit part of the full specification of the path of a file, when giving a filename as a parameter to a command in the Command Processor, so also is it often possible to omit part of the full specification of a *path* (or *directory name*), when issuing a command such as `cd`, `md`, or `rd` that expects a directory name as a parameter. Any missing parts of the directory name will be filled in from the current path.

Thus if the current drive is *m:* and the current path is *\img\*, the command `md tools` is equivalent to `md m:\img\tools`. Likewise `cd tools` is equivalent to `cd m:\img\tools` and `rd tools` is equivalent to `rd m:\img\tools`.

A trailing backslash can generally be used to clarify that a name is a directory rather than a file.

## WildcardsXE "Wildcards"§tc "Wildcards" ¶ 3§

As in DOS, the wildcards `?` and `*` may be used in the file name and/or the extension components of a file specification, where `?` represents any single character and `*` represents zero or more characters.

The use of `*` differs from its use in DOS in that, on the *Workabout*, it is possible to use, for example:

```
a*b.*
```

to select all files whose file name starts with *a* and ends with *b*. In DOS, `a*b.*` has the same effect as `a*.*`.

## The requirements of generalitytc "The requirements of generality" ¶ 3§

Evidently, although some of the syntax of command such as `md` and `rd` is similar to that of corresponding commands in MS-DOS, in other aspects the requirements of the *Workabout* Command Processor syntax for these functions may be found more punitive than in MS-DOS.

In fact, the extra restrictions stem from a central design feature of the EPOC operating system - the requirement for applications to be able directly to address files on a remote computer, where the remote filing system may well be *other than MS-DOS*. Alternative remote filing systems that need to be borne in mind, as well as just MS-DOS, include Unix, Vax VMS, and the Apple Macintosh operating system.

Thus if the *Workabout* is connected to an Apple Macintosh computer, the following could be entered at the command line:

```
dir rem::hd40:Workdevp:stock
```

Accordingly, the *Workabout* Command Processor does not simply approach filenames and path specifications in terms of presence or absence of backslashes. For example, were the Command Processor to insert a backslash at the end of the above command, "on behalf of the user", this would, most decidedly, *not* be what the user intended. Instead, the approach is much more general, in terms of the five part breakdown of filename specifications discussed two sections previously.

It should be noted that Psion computers allow the use, for example, of the "+" character in filenames, which is not allowed in MS-DOS for instance. For portability between filing systems and applications on different manufacturers machines it is recommended that only letters and numbers are used in filenames. At the very least filenames should be restricted to the MS-DOS naming convention. The allowed characters in an MS-DOS filename are:

A-Z a-z 0-9 \$ % ' - @ { } ~ ` ! # ( ) &

MS-DOS filenames are case insensitive, (upper and lower case letters are treated the same).

Different DOS versions handle accented characters differently. It is recommended that accented characters are not used in filenames.

Filenames that are keywords, the names of ports or device drivers, etc. on the *Workabout* or any remote system, should also be avoided, e.g. COM1, LPT2, etc.

Similarly, the *Workabout* provides only limited support for the syntax of "double dot" (for the parent directory) and "single dot" (for the current directory).

This extra discipline has its occasional drawbacks. However, the advantages that it brings with it are an important part of the vital *inter-connectable* feature of the *Workabout*.

### Default location of filesXE "Files:default location"\$XE "Default:location of files"\$XE "Location of files, default"\$Xc "Default location of files" \l 3§

The default location of files of particular types is given in the table below. You should try to store any files of your own within this existing directory structure. For example, all new printer drivers should be stored in the \wdr directory.

File type	Directory	Extension
Application program	\app	.app
Batch	\btf	.btf
Communications 'receive'	\opl	n/a

Communications script (editable)	<i>\scr</i>	<i>.scr</i>
Communications script (translated)	<i>\sco</i>	<i>.sco</i>
Communications 'transmit'	<i>\opl</i>	<i>.opl</i>
Database	<i>\dat</i>	<i>.dbf</i>
OPL program (source)	<i>\opl</i>	<i>.opl</i>
OPL program (translated)	<i>\opo</i>	<i>.opo</i>
OPL application (OPA)	<i>\app or \img</i>	<i>.opa</i>
OPL data files (unless specified by the OPA)	<i>\opd</i>	<i>.odb or .pic</i>
Printer drivers and program editor (Word application) templates	<i>\wdr</i>	<i>.wdr or .o</i>

Program image file	<i>\img</i>	<i>.img</i>
Spreadsheet	<i>\spr</i>	<i>.spr</i>

---

## Alphabetical listing "Alphabetical listing" ¶ 2§

### Notation "Syntax notation, commands" § "Command:syntax notation" § "Notation" ¶ 3§

This list of commands uses the following syntax:

COMMAND supplied-parameter [optional-parameter | optional-parameter]

or:

COMMAND [optional-parameter] {supplied-parameter | supplied-parameter}

Items shown in square brackets ( [ ] ) are optional. To include optional information, type only the information within the brackets. Do not type the square brackets themselves.

The use of the vertical bar ( | ) symbol between parameters means that one parameter OR the other parameter can be used, (but not both). Type only one parameter. Do not type the vertical bar itself.

If there are alternatives but one or other **must** be included, the options are enclosed in curly brackets ( { } ), if required for clarity. Type only one parameter. Do not type the curly brackets themselves.

Shortened versions of commands are **not** available. In the above (generalised) example, COM would not be an acceptable shortened form of COMMAND. The Workabout Command Processor thus differs from the Psion HC Command Processor which does allow command abbreviations.

Commands can be typed in any combination of lower and upper case. For example, except where clearly stated to the contrary below, pairs of command such as wnot on and wnot ON are completely equivalent.

Commands must be separated from their options by inserting a space character.

Default values may be assumed if some options are not supplied. Default values of particular commands are given in the individual command descriptions which follow.

Note: the following list actually contains entries that are not really commands of the Command Processor, in the strict sense, but are just the name of a program in the ROM, for example link . However, this distinction may seem irrelevant to the user, and so, for convenience, these commands are listed as well. **DOS calls these external commands.**

### ATTRIB "Command:ATTRIB" § Set or clear file attributes § "Set:file attributes (A

ATTRIB [system:][drive:][path]filename[.extension] [+a-a] [+s-s] [+h-h] [+r-r]

Sets or resets the archive (a), system (s), hidden (h) and/or read-only (r) attributes of a file.

For example, attrib list.dat -a +s clears the archive attribute and sets the system attribute of *list.dat*, without altering its hidden or read-only attributes.

For each of h, s, a, and r, a prefix of - clears the corresponding attribute, and a prefix of + sets it. The four attributes can be specified in any order, and any combination of the four bits can be set or cleared at once. If all four attributes are omitted, attrib displays the existing settings.



If the filename is also omitted then the attributes of **all** files in the current (or specified) path are displayed.

The `attrib` command **does** accept a wildcard specification.

If the archive flag (a) is set it means "this file must be archived". Typically the archive flag is cleared after the file has been backed up (archived).

Note that the `DIR` command includes the attributes of files as part of its display.

### **CALCxe "Command:CALC"\$Run the calculator applicationXE "Run:calcu**

[START] CALC

Run the Calculator application, Calc.

If preceded by `START`, Calc will be run asynchronously.

#### **Note**

Help on using the Calculator application is available by pressing Shift-Esc.

For full information on using the Calculator application, see the *Workabout User Guide*.

### **CALLxe "Command:CD"\$Call a batch file from inside another batch fileXE (CALL)" \I 3\$**

[CALL] [[system::][drive:][path]batchfile[.extension] [batch-parameters]

Calls a batch file from inside another batch file.

Omitting the `CALL` command and just using the file specification "chains" the named batch file, and does not return to the current batch file on completion.

### **CDxe "Command:CD"\$Display or alter current directoryXE "Display:current directory (CD)" \I 3\$**

CD [system::][drive:][path]

Changes to a different path, or (if `path` is omitted) displays the current path.

The *Workabout* maintains one path per local drive, plus a "current drive".

**Note that** `system::` **can only be** `loc::`.

For example, to change the current directory from `\work\product\` to `\work\admin\`, type:

```
cd \work\admin
```

To move to a directory below the current one, only the path from the current directory needs to be entered. So to change from `\work\admin\` to `\work\admin\forms\`, the following command could be used:

```
cd forms
```

The trailing backslash in the above commands can be omitted. Thus `cd forms` instead of `cd forms\` will do the same thing.

Use `CD [drive:]` to change to the root directory of a drive, or of the current drive if `drive:` is omitted.

For example, type:

```
cd b:\
```

to change to the root directory of `b:`.

Typing `cd b:` changes nothing and displays the current path for *b:*.

To change drive use:

drive:

where drive: is *a:*, *b:*, *c:*, etc.

## **CHDIRxe "Command:CHDIR"\$Display or alter current directoryXE "Display:current "Display or alter current directory (CHDIR)" \| 3§**

CHDIR [system:][drive:][path]

Changes to a different path, or (if *path* is omitted) displays the current path. This command is exactly the same as `CD`.

## **CLSxe "Command:CLS"\$Clear the screenXE "Screen:clear (CLS)"\$XE "Clear:the s**

CLS

Clears the screen, leaving the `>` prompt in the top left corner.

## **COMMSxe "Command:COMMS"\$Run the communications applicationXE "Run:the**

[START] COMMS [system:][drive:][path][filename[.extension]]

Run the communications application, Comms.

If preceded by `START`, Comms will be run asynchronously.

If no file specification is provided, and the current drive is *m:*, (and *comms.sco* does not already exist), a dialog is presented asking:

Create "M:\SCO\COMMS.SCO"?

If 'Y' is then pressed the file *m:\sco\comms.sco* will be created, along with a file *m:\scr\comms.scr*, for later use. Note that files with the extension *.scr* are editable communications scripts files, and files with the extension *.sco* are translated communications script files. The user is then presented with the terminal emulation screen.

Otherwise, if *comms.sco* does already exist, then no dialog is presented. Note that it does not matter whether *comms.scr* exists or not.

If any system, drive or path is provided, but no filename, the *comms.sco* file will be placed in the specified location. The default directory for translated communications script files (extension *.sco*) is *\sco\*, and it is recommended that this is where all files of this type are kept. Note that the default directory for untranslated (editable) communications script files (extension *.scr*) is *\scr\*, and keeping all files of this type in their directory is also recommended.

If a filename is given, a *.sco* file and a *.scr* file of that name will be created (as above) depending on whether they currently exist or not.

An extension to the filename may be provided that differs from *sco*, but this is not recommended, since it would make identification of the file type difficult.

### **Note**

Help on using the Communications application is available by pressing Shift-Esc.

For general information on using the Communications application, see the *Workabout User Guide*, and for full details refer to the *Psion Series 3 3Link (RS232)* manual.

## **COPYxe "Command:COPY"\$Copy file(s)XE "Copy file(s) (COPY)"\$XE "File**

COPY source\_filespec destination\_filespec [/s] [/y]

Copies one or more files or directories, possibly changing their names in the process.

If the /s flag is used then all subdirectories of a specified source directory will also be copied.

The /y switch allows overwriting without confirmation.

Any part of the target filename that is not specified, (for example, the extension), and which cannot be filled in from corresponding parts in the current drive and path, is taken from the corresponding part of the first pathname.

If the destination filename is not specified then any existing file of the same name as the source file will be overwritten. A "File exists" **dialog** appears **before** this has happened, **unless /y is used**. A similar thing happens if the destination filename is the name of an existing file.

The wildcards \* and ? can be used to copy multiple files.

For example:

```
copy fred.* a:\jim.*
```

copies all files such as *fred.btf* from the current directory into the root of *a:\*, renaming them (e.g. to *jim.btf*) in the process.

Several files **cannot** be combined into one file by using COPY with wildcards.

As a possibly surprising example, if the current **drive is m: and the current** path is \, the command

```
copy a:\file.lis file.old
```

has the effect of copying the named file to *m:\file.old*.

As files are copied, the names of the **(source)** files **copied** are listed on the screen.

A file cannot be copied onto itself. If an attempt is made to do this, the COPY command quits, and an error message such as the following is displayed:

```
Copy failed
```

## **DATAxe "Command:DATA"\$Run the database applicationXE "Run:the da**

[START] DATA [system::][drive:][path][filename[.extension]]

Run the database application, Data.

If preceded by START, Data will be run asynchronously.

If no file specification is provided, and the current drive is *m:*, (and *data.dbf* does not already exist), a dialog is presented asking:

```
Create "M:\DAT\DATA.DBF"?
```

Otherwise, if *data.dbf* does already exist, then it will be presented directly for editing.

If any system, drive or path is provided, but no filename, the *data.dbf* file will be placed in, (or retrieved from), the specified location. The default directory is *\dat\*, and it is recommended that this is where all database files are kept.

If a filename is given, a *.dbf* file of that name will be created (as above) and/or presented for editing depending on whether it currently exists or not.

An extension to the filename may be provided that differs from *.dbf*, but this is not recommended, since it would make identification of the file type difficult.

For general information on using the Database application, see the *Workabout User Guide*.

This application may be useful for quickly looking at or manipulating OPL data files. There are, however, certain restrictions and differences that must be borne in mind. See the *Database files* chapter of the *SIBO OPL Technical Reference* manual for further details.

### Note

Help on using the Database application is available by pressing Shift-Esc.

## **DATE** **Command:DATE** **\$Display date** **XE "Display:date (DATE)"\$XE "Date:disp**

DATE

Displays the current date in the default format, or the format specified by the most recent SETDEF or menu command.

The default display format is:

dd/mm/yy

For example 12/10/95 will be displayed if the current system date is 12 October 1995.

If the day of the month or the month number itself has only a single digit, a leading zero is displayed. The first day of January 1996 is displayed as:

01/01/96

This different from MS-DOS.

Use the Time and Date option on the Time menu to change the date.

Use the SETDEF command or the Formats option of the Time menu to change the format in which the date is displayed.

**DELxe "Command:DEL"\$Delete file(s)XE "Delete:file(s) (DEL)"\$XE "File(s)**

DEL [system:][drive:][path]filespec [/s] [/y]

Deletes the specified file or files.

If the /s flag is used, all subdirectories and **the files in them** are deleted.

If the /y flag is used, no confirmation is asked for. This is intended for use in batch files. Note that, like in MS-DOS, confirmation is not asked for if a **single** file is being deleted. Unless /y is used, confirmation is always asked for if **all files in any directory** are being deleted.

To delete more than one file at a time, the wildcards \* and/or ? can be used. Alternatively, the following deletes all files in the directory *temp* (in the current path):

```
del temp\
```

The trailing backslash may be omitted, in which case a dialog is presented: "Delete all files in directory?". The Y key should then be pressed to carry out the deletion.

As files are deleted, the names of the files deleted are listed on the screen.

The ERASE command is exactly the same as DEL.

See also the command RD, which deletes directories.

**DIRxe "Command:DIR"\$Full directory listingXE "Full directory listing (DIR**

DIR [system:][drive:][path][filespec] [/s] [/b]

Lists all the specified files in a directory, together with their sizes, the time and date of their last modification, and their attributes.

The listing pauses automatically after each screen of information, prompting the user to press enter to continue, (except in batch files).

The wildcards \* and ? can be used in the file specification.

The /s flag causes all subdirectories to be listed as well.

The /b flag causes bare (filenames only) output to be used.

**ECHOxe "Command:ECHO"\$Display message, set or display echo mode> display echo mode (ECHO)" \I 3§**

ECHO [[text]][ON | OFF]]

Displays the message *text*, or sets the echo mode in batch files, or displays the current echo mode.

```
echo Processing
```

Displays the message *Processing* on the next line of the screen.

```
echo on
```

Switches the echo mode in batch files to ON.

echo off                      Switches the echo mode in batch files to OFF.

echo                          Displays the current echo mode in batch files, e.g.:

Echo is OFF **or**

Echo is ON

**EDITxe "Command:EDIT"\$Run the **text** editorxe "Editor:text file, run (EDIT)"\$xe "Text (EDIT)" \l 3\$tc "Run the text editor (EDIT)" \l 3\$**

EDIT [system:][drive:][path]filename[. extension]

Runs the **text** editor program.

The three different editors for OPL programs, communications scripts and batch files, are implemented as aliases of the Word application.

A filename must be given (no default names are provided).

If no extension is given then an alias of Word that behaves as a batch file editor is started, and the extension defaults to *.btf*. In this instance there are no Translate and Run menu options available, since they are not relevant.

If the filename is given an extension of *.opl* then Word behaves as an OPL program editor. If the filename is given an extension of *.scr* then Word behaves as a communications script editor. In both these instances the Translate and Run menu options are enabled. **Note that translation is different for *.scr* and *.opl* files.**

If a path is not given then *\prog\* or *\scr\* or *\btf\* is assumed as the directory for an OPL program (*.opl*), communications script (*.scr*) and batch file (*.btf*) respectively.

If any other extension is given for the filename and no path is given then the file will be put in *\btf\*.

If you supply a partial path, the Command Processor will try to use the results of your last CD command on the relevant drive. This may be thought of as "DOS rules".

If a filename **is not** supplied the error message "Too few parameters" is displayed.

If a filename **is** given, but no extension or preceding backslash or directory, you will end up editing *\btf\filename.btf* on the current drive:

- For example, if the current drive is M: and a file called *filename.btf* does not already exist, then a dialog is presented, asking:

Create "M:\BTF\FILENAME.BTF"?

If Y is then pressed the editor will display a blank file of name *filename.btf*, ready for editing.

- If a file called *\btf\filename.btf* does already exist, then no dialog is presented, and the existing file *filename.btf* is presented for re-editing.
- The existence of *.opl* or *.scr* files with the same name does not affect the situation

described above.

If, however, you include any backslash, then 'DOS rules' will prevail:

- For example, if the current directory is *a:\fred* and you enter:

you will edit *a:\fred\tom\filename.btf*.

- The default file extension is always *.btf*, regardless of which directory is current or specified. If, for example, the current directory is *m:\opl* and you enter:

EDIT \filename

you will edit *m:\opl\filename.btf*, even though you are in the *\opl* directory.

Similar dialogs and messages are presented for files with *.scr* and *.opl* extensions.

When saved, or the editor is exited, files are stored in the specified directory.

### Note 1

The `EDIT` command starts an alias of the Word application. When `EDIT` (or `START EDIT`) is used the process started will appear in an `LPROC` or `LSEG` listing as an instance of Word. As well as by using `EDIT` directly from the Command Processor or from a batch file aliases of Word can be started from one of the editor icons in the System Screen.

The first time that `EDIT` is used to run the batch file editor, the program editor or the communications script editor, a *\wdr\* directory will be created (if it does not already exist). This is for printer drivers and templates. Furthermore, the first time the program editor is used the default template *default.o* is stored in this directory.

### Note 2

Help on using the Editors is available by pressing Shift-Esc. All the editors function in the same way.

For further information on the OPL Program Editor see the *Workabout User Guide* and for detailed information see the *Workabout Programming Guide*.

For further information on the Batch File Editor see the *Workabout User Guide*.

For further information on the Communications Script Editor see the *Workabout User Guide* and the *Psion Series 3 3Link (RS232)* manual.

## ERASE "Command:ERASE"\$Delete file(s)tc "Delete file(s) (ERASE)" \l 38

ERASE [system::][drive:][path]filespec [/s] [/y]

Deletes the specified file or files.

If the */s* flag is used, all subdirectories and **the files in them** are deleted.

If the */y* flag is used, no confirmation is asked for. This is intended for use in batch files. Note that, like in MS-DOS, confirmation is not asked for if a **single** file is being deleted. Unless */y* is used, confirmation is always asked for if **all files in any directory** are being deleted.

To delete more than one file at a time, the wildcards *\** and/or *?* can be used. Alternatively, the following deletes all files in the directory *temp* (in the current path):

```
erase temp\
```

The trailing backslash may be omitted, in which case a dialog is presented: "Delete all files in directory?". The Y key should then be pressed to carry out the deletion.

As files are deleted, the names of the files deleted are listed on the screen.

The command `DEL` is exactly the same as `ERASE`.

See also the command `RD`, which can delete directories.

### **ERRLEVEL**`x` "Command: ERRLEVEL"\$Display error level state`t`c "Display error le

`ERRLEVEL`

Display the error level state, e.g.:

```
Errorlevel is FALSE
```

It reports whether the last command or process to finish left `ERRORLEVEL` set as `TRUE` or `FALSE`. This may be useful when debugging batch files.

### **ERRORLEVEL**`x` "Keyword, ERRORLEVEL"\$Error level value`t`c "Error level value (E

The keyword `ERRORLEVEL` is not a command. It is the status of the last command that was executed. It holds the error level which is a Boolean value, `TRUE` or `FALSE`. Unlike DOS, Command Processor commands set `ERRORLEVEL` as applicable. The general use of `ERRORLEVEL` is in an `IF` statement:

```
IF [NOT] ERRORLEVEL command
```

Thus, for example, within a batch file, you can test for a directory, as follows:

```
CD \INC
IF ERRORLEVEL MD \INC
REM If there is no INC directory, (CD set ERRORLEVEL as TRUE), make one
CD \INC
```

### **EXIT**`x` "Command: EXIT"\$Terminate the Command Processor/Exit level`t`c "Termin "Exit:batch processing level (EXIT)"\$`x` "Batch file: exit processing level (EXIT)"\$

`EXIT`

Present a dialog offering options to terminate the Command Processor or cancel the command.

On confirmation of an `EXIT` command typed directly into the Command Processor (or of the selection of the Exit option from the Special menu) the Command Processor is terminated. If there is no other task running on the machine, the *Workabout* will then automatically bring the Startup Shell process to foreground, as explained in the chapter *Introduction to the Workabout*. If one or more other tasks are running, the machine will switch tasks to whichever running application was last used.

If the `EXIT` command is executed from a (nested) batch file, all levels of batch processing are terminated immediately and the confirmation dialog presented. On confirming the `EXIT` command, the Command Processor terminates in the same way as when `EXIT` is typed directly into the Command Processor.

### **FILES**`x` "Command:FILES"\$List open files`t`c "List open files (FILES)" \| 3\$`x` "Open

`FILES [system::][drive:]`

Lists all open files on the specified drive.

If the drive is not specified, open files on the current drive are listed.



The program using each file, the process number and the full filename including the path are given.

**FORxe "Command:FOR"\$Run a command for the files in a settc "Run a co  
(FOR)"\$xe "Run:a command for the files in a set (FOR)"\$**

FOR %[%]variable IN (set) DO command [parameters]

Run a command for each file in a specified set of files.

The replaceable parameter %variable is used by the FOR command to hold the name of each file as it is being processed. The extra % is needed in batch programs.

The set of files is specified by set. The wildcards \* and ? can be used. The parentheses are always required.

The command to carry out for each file in the set is specified by command. The parameters or switches for the specified command are given by parameters, if the command needs any.

To use the FOR command in a batch file, specify %%variable instead of %variable.

For example, to asynchronously run all application files in the a:\app\ directory:

```
for %f in (a:\app\*.app) do start %f
```

From within a batch file, this same sequence of operations would be performed by the line:

```
for %%f in (a:\app\*.app) do start %%f
```

**FORMATxe "Command:FORMAT"\$Format and re-label local volumetc "For  
label (FORMAT)"\$xe "Volume:local, format and re-label  
(FORMAT)"\$**

FORMAT [system:][drive:][volname]

Formats a local volume i.e. a RAM or Flash SSD in drive A: or B:, or the internal drive M:RAMDRIVE.

It also re-labels the volume with the optional "name".

The command detects the type of SSD and places the appropriate format information onto the disk. This information differs for Flash and RAM SSDs.

For example, format a: will format the SSD in drive A: , retaining the existing label, if any.

The command, format b:data will format the SSD in drive B: and label it as "DATA".

If drive: is omitted, a dialog appears asking if you want to format the current drive.

The mere fact that there are read-only files on an SSD or the internal drive will not prevent it from being formatted. However, if an SSD has the write-protection switch set, it will not be possible to format it.

The FORMAT command will fail if there is no SSD in the specified drive. In this case, the FORMAT request will fail with the error message "Format failed. Not ready".

Another reason for FORMAT being disallowed for a disk would be if there are any open files on it. In this case, the FORMAT request will fail with the error message "File or device in use".

If an SSD needs to be labelled at a later date then the `LABEL` command may be used.  
To find the current label on a volume use the `VOL` or `DIR` command.

Note: the label in this case labels the SSD volume. It is not the same thing as the label mentioned in the description of the `GOTO` command below.

## **GOTOxe "Command:GOTO"\$Jump to label in batch filetc "Jump to label in batch fi**

`GOTO label`

Jump to the label `label` in a batch file.

Note: the `label` in this case is a label at the start of a line in a batch file (it is preceded by a colon, `:`). It is not the same thing as the label mentioned in the `FORMAT` and `LABEL` command descriptions.

## **HELPe "Command:HELP"\$List commands or get helptc "List commands or get he**

`HELP [command]`

List the available commands, or supply help on a specific command.

For example entering:

`HELP label`

returns the following message:

`GOTO label`  
`Jump to label in a batch file`

Press Shift-Esc for access to the Help System, or Ctrl-Esc for the Help Index.

## **IFxe "Command:IF"\$Run command conditionallytc "Run command conditionally (IF**

`IF [not] {ERRORLEVEL | string1==string2 | EXIST filespec} command [command_parameters]`

Conditionally run a command, or executable file, depending on one of three possible specified conditions.

- The value of `ERRORLEVEL`., for example:

`IF ERRORLEVEL batchfile.btf`

Note that `ERRORLEVEL` can only be `TRUE` or `FALSE`. The DOS syntax:

`IF ERRORLEVEL n ...`

is not supported.

- Whether two strings (`string1` and `string2`) are equivalent, for example:

`IF ?%1==? DEL *.tmp`

- Whether a file specification `filespec` exists, for example:

`IF EXIST tmp.tmp DEL tmp.tmp`

or:

`IF EXIST a: COPY m:\btf\ a:\btf\`

The file specification can consist of a system and double colon, drive letter and colon, a directory name, a filename, or a combination of these.

If `not` is omitted, it performs `command` if the following specified condition is true.

If `not` is included, it performs `command` if the following specified condition is false.



**KILLxe "Command:KILL"\$Kill a processtc "Kill a process (KILL)" \l 3\$xe "Kill a proce**

KILL procname [/y]

Kills all the processes found matching the specification in `procname`.

**Note that this is intended for emergency use only.**

If the `/y` flag is used, no confirmation is asked for. This is intended for use in batch files **and should be used with extreme caution.**

To kill a specified instance of a number of running tasks, all with the same name, the exact process name must be found out and used. For example:

```
kill job.$09
```

or

```
kill job.$14
```

Use the `LPROC` command to give the full process names of all current processes.

If there are no processes matching the specification, a notifier stating:

```
Unknown process "Procname"
```

will be presented.

Note that `KILL` should only be used as a last resort, as it does not allow the process to tidy up before exiting. This is a problem, for example, with the Link application which starts a number of sub-processes that will not be shut down if Link is stopped with `KILL`. To shut down a process, `STOP` should normally be used in preference to `KILL`.

A confirmation dialog is presented if a process matching the specification is found, reminding the user that `KILL` is for emergency use only:

```
Kill "Procname"  
(Emergency only)
```

**Note:**

The `KILL` command is designed for emergency use with badly behaved, **perhaps partially developed**, user-written programs. Built in Psion applications should be stopped using the options available from their menus or those of the System Screen, **or by the `STOP` command.**

**LABELxe "Command:LABEL"\$Add/alter disk volume labeltc "Add/alter disk volume add or alter (LABEL)"\$xe "Label:volume, add or alter (LABEL)"\$**

LABEL [system:][drive:][volname]

Labels the volume in the specified `drive` with the label `volname`.

If the drive is omitted then the current local volume (i.e. an SSD in the current drive, A: or B:; or the internal RAMdrive M:), will be re-labelled with the label `volname`.

If the name `volname` is omitted then a "Delete label" dialog will appear (unless the volume in the specified `drive`, or current drive, has no label).

For example:

```
label a:backup
```

Will re-label the SSD in drive A: as "backup".

If the command `label a:` is now used then the dialog 'Delete label "A:BACKUP" ' will be displayed. Pressing Y then deletes the label, pressing N or Esc abandons the command.



**LINKxe "Command:LINK"\$Start LINK programtc "Start Link program (LINK)" \l 3\$xe**

LINK [-b<baud>] [-p<port>]

Starts the LINK communication software on the *Workabout*. It is always run asynchronously, so **START** is not needed.

If the command line is empty, the LINK software is started with the default settings, or those used on the last occasion that LINK was started, **whether by this command or via the dialog in the System Screen or Command Processor menus.**

Possible values of `baud` range from 19200 and 9600 all the way down to 110, 75, and 50, with all common baud rates in between being supported. In the absence of command line parameters, or if `port` is specified but not `baud`, `baud` defaults to 19200.

The only time it is necessary to specify `port` is if there are serial expansion devices in the *Workabout*.

- `-p1` means to use the expansion RS232 port, (port A)
- `-p3` means to use the LIF socket (LIF Converter with attached 3Link inserted) as an RS232 serial port (port C)

Otherwise, if `port` is not specified, the LINK software simply uses the first available port in alphabetical order.

Other parameters are also possible, but are omitted from the present description. See the chapter *MCLINK*, *MCPRINT*, and *SLINK* in the *Tools Reference* manual. Just typing `link` should normally suffice.

To terminate the LINK software at some later stage, type `stop link`.  
A Yes/No dialog will be presented asking:

Stop "Link"?

`y` should then be pressed to stop LINK.

To discover whether or not LINK software is running, type `lproc link`.

If the `link` command is issued while LINK is already running, no harm will be done.

See *Connecting to other computers* in the chapter *Introduction to the Workabout*, for more details.

**LLDEVxe "Command:LLDEV"\$List logical device driverstc "List logical device driver"**

LLDEV [device\_spec]

Lists all specified logical device drivers. The list includes all ROM-resident device drivers, as well as external ones that are currently loaded.

If `device_spec` is omitted, it defaults to `*`, and all logical device drivers will be listed.

Logical device drivers are hardware-independent.

For example, entering `lldev con` displays

```
Logical devices matching: con
CON (unlimited units)
Matches found 1
```

The value given for `units` is the number of channels that the corresponding logical device driver can support. For the console device (`CON:`) an unlimited number of channels can be opened.

A full list of the default logical devices, for an unexpanded machine, is given below

<b>Driver</b>	<b>Units</b>	<b>Driver description</b>
WSX	0	O/S use only
CON	Unlimite	Console
EM\$	Unlimite	Floating Point Emulator
TCK	4	Tick driver
NCP	Unlimite	Network Controller; supports LINK
FRC	1	Free Running Counter
PMX	1	Pack Multiplexer
CRD	Unlimite	Cradle
WLD	Unlimite	World support (not accessible unless an application using it)
ALM	Unlimite	Alarm support (not accessible unless an application using it)
DBG	1	Debug support
XMD	Unlimite	Xmodem/Ymodem
LLC	Unlimite	Logical Link Layer; supports LINK
MAS	Unlimite	Link Media Access Control; supports LINK
MCR	3	Magnetic card reader
PAR	3	Parallel port
TTY	2	Serial port; supports LINK
SND	1	Sound
TIM	Unlimite	Timer
TXT	Unlimite	File system
FIL	Unlimite	File system

## **LPDEVxe "Command:LPDEV"\$List physical device drivers\$tc "List physical**

LPDEV [device\_spec]

Lists all specified physical device drivers that match the supplied specification. The list includes all ROM-resident device drivers, as well as external ones that are currently loaded.

If `device_spec` is omitted, all physical device drivers will be listed..

Physical device drivers are hardware dependent.

For example, entering `ldev fsy` displays:


```
Devices matching: fsy
FSY.REM
FSY.LOC
FSY.ROM
Matches found 3
```

listing the three ROM-resident filing system device (`fsy`) drivers - for `rem::`, `loc::`, and `rom::`.

A full list of the default physical devices, for an unexpanded machine, is given below

<b>Driver type</b>	<b>Driver description</b>
FSY.REM	Filing system: <code>rem::</code>

LOC.TYL	RCom support
TTY.SRX	Serial port - ASIC4 (e.g. 3Fax); supports LINK
FSY.LOC	Filing system: <i>loc::</i>
FSY.ROM	Filing system: <i>rom::</i>
LOC.TYM	Internal "drive"
LOC.TY2	SSD support
LOC.TY1	SSD support
LOC.TY0	SSD support
TTY.HSS	Serial port - High Speed Serial; supports LINK
TTY.UAR	Serial Port - ASIC5 (e.g. 3Link); supports LINK





**LPROCxe "Command:LPROC"\$List processes<sub>tc</sub> "List processes (LPROC)"**

LPROC [process\_spec]

Lists information about all specified processes. The information listed is:

- The full process name (in the form *procname.\$07*).
- The size, in bytes, of the process data segment (given in hexadecimal).
- The current state of the process.

The total number of matches found is then displayed.

If `process_spec` is omitted, information is listed for all current processes.

Possible values of the state of the process are:

CURRENT	The process is currently receiving CPU time.
READY	The process has some events ready to process, as soon as the CPU is given to the process by the multi-tasking scheduler.
DELTA	The process is "sleeping" (e.g. as a result of calling <code>PAUSE</code> with a <i>positive</i> parameter).
SUS	The process has been suspended.
SEM	The process is waiting for some event to happen.

Additionally, the text `WSusp` will be displayed if the process is *waiting* to be suspended.

For example, entering `lproc` may produce the display:

```
Processes matching: *
SYS$NULL.$01 0300 READY
SYS$MANG.$02 0DA0 SEM
SYS$FSRV.$03 1320 SEM
SYS$WSRV.$04 4DC0 SEM
SYS$SHLL.$05 14C0 SEM
SYS$CMDP.$06 4BC0 CURRENT
Matches found 6
```

As a further example, entering `lproc sys$shll` may produce the display

```
Processes matching: sys$shll
SYS$SHLL.$05 14C0 SEM
Matches found 1
```

One common use of the `lproc` command is to check whether LINK software is currently running: `lproc link`.

Notes:

For processes which are aliases of another program, using `START procname` (or `procname` on its own) will start a process that appears to `LPROC` as an instance of the base process. For example, all text and program editors are aliases of the Word application. In consequence processes started with `EDIT` or `START EDIT` all appear as instances of Word in an `LPROC` listing.

Note also that Sheet is the user-visible name of Sh3. A process started with `START SHEET` or with `SHEET` will appear as an instance of Sh3 in the `LPROC` listing.

**LSEGxe "Command:LSEG"\$List segments<sub>tc</sub> "List segments (LSEG)" \I 3\$<sub>x</sub>**

LSEG [process\_spec]

Lists all memory segments currently in use by the specified process(es).

If `process_spec` is omitted, the listing will include the memory segments used by all currently running processes.

The information listed about each memory segment is:

- Its segment address.
- Its size in paragraphs (one paragraph is sixteen bytes).
- Its access count.

Values are displayed in hexadecimal.

The total number of matches found is then displayed.

At the end the display, the total size in paragraphs of all the free segments is given (this gives the same value, when converted into kilobytes, as the `MEM` command).

For example, entering `lseg` may produce the display:

```
Segments matching: *
SYS$NULL.$01 046E 0030 01
SYS$MANG.$02 049E 00DA 01
SYS$WSRV.LDD 0578 00E3 01
SYS$FSRV.$03 065B 01B7 01
SYS$WSRV.$04 0812 0560 01
SYS$SHLL.$05 0D72 014C 01
SYS$CMDP.$06 0EBE 04BC 01
Matches found 7
Total free segments = 6CF0
```

#### Notes:

For a process that is an alias of another built-in program (`EDIT` or `SHEET`), using `START procname` (or `procname` on its own) will start a process that appears to `LSEG` as an instance of the base process. This does not apply to `.als` files.

For example, all text and program editors are aliases of the Word application. In consequence processes started with `EDIT` or `START EDIT` all appear as instances of Word in an `LSEG` listing.

Note also that Sheet is the user-visible name of Sh3. A process started with `START SHEET` or with `SHEET` will appear as an instance of Sh3 in the `LSEG` listing.

**MDxe "Command:MD"\$Make directory<sub>tc</sub> "Make directory (MD)" \I 3\$<sub>xe</sub> "Mak**

MD [system::][drive:]path

Makes a directory.

It is possible to omit the trailing backslash from the path specification.

The following commands both create a directory named `\work\` in the root directory of the current drive:

```
md \work\
md \work
```


The command `MKDIR` (which is identical) is also available.

**MEMxe "Command:MEM"\$Display free memorytc "Display free memory (MEM)" \l 3**

MEM

Displays the amount of free RAM in kilobytes that is available to programs.

Note that this may differ from the amount of bytes free on *m:*, as reported by a `DIR` or `VOL` command. The discrepancy is because some parts of internal memory are reserved for code and data segments; not all of it can be allocated to the contents of *m:*, and vice-versa.



**MKDIRxe "Command:MKDIR"\$Make directorytc "Make directory (MKDIR)"**

MKDIR [system:][drive:]path

Makes a directory.

It is possible to omit the trailing backslash from the path specification.

The following commands both create a directory named `\work\` in the root directory of the current drive:

```
mkdir \work\
mkdir \work
```

The command `MD` (which is identical) is also available.

**PAUSExe "Command:PAUSE"\$Suspend batch file processingtc "Suspend**

PAUSE [text]

Suspend batch file processing.

The message `text` is displayed (if given).

The prompt

```
Press Enter to continue...
```

is shown.

When Enter is pressed batch file processing continues by executing the next line of the current batch file.

**QUITxe "Command:QUIT"\$Exit current batch file onlytc "Exit current batch**

QUIT

Terminate the current batch file. If batch files are nested, batch processing will continue at the line following that which contains the call to the batch file that executed `QUIT`.

The command has no effect if typed directly into the Command Processor.

**RDxe "Command:RD"\$Remove directorytc "Remove directory (RD)" \| 3\$xe**

RD [system:][drive:]path [/y]

Deletes a directory, including any files in it (and subdirectories).

If the `/y` flag is used, no confirmation is asked for. This is intended for use in batch files.

Note that, in contrast to MS-DOS, there is no requirement to delete all the files in a directory before removing the directory.

In another difference from MS-DOS, it is perfectly possible, in the *Workabout* Command Processor, to remove the directory where the current path is. All that will happen is that subsequent commands such as `dir` may fail until such time as the current path is changed.

As files and directories are deleted, their names are listed on the screen.

The `rd` command does not accept a wildcard specification.

The identical command `RMDIR` is also supported.



## REASONxe "Command:REASON"\$Get the cause of the last system shutdown (REASON)"\$

REASON

Displays a numeric code indicating the cause of the last system shutdown, as follows:

- 0        The system started up for the first time (or after a period during which all power had been removed, including the Lithium back-up battery).
  
- 1        xe "Hardware interrupts"\$The hardware forced a shut-down because the voltage got too low. This should not happen because the system software receives a non-maskable interrupt if the voltage drops below a certain threshold (but not low enough for the hardware to force a shut-down). This interrupt code automatically switches the hardware off. However, if the clean-up takes too long (because of a poorly designed device driver, for example) the hardware will force the machine off before the interrupt completes - in which case REASON returns 1. The environment variables and the contents of M: are preserved.
  
- 2        The user pressed the recessed reset button. This is not relevant to the Workabout, since it does not have a reset button.
  
- 3        *Either:*  
          The user pressed Psion-Ctrl-Del to perform a soft reset. The environment variables and the contents of M: will have been preserved.  
          *Or:*  
          This reason is also given if the system was reset because a serious fault occurred while executing code in the operating system kernel. This could result from three things:
  - A bug in the operating system or a system process.
  - A program bug that managed to overcome the operating system's defences.
  - A hardware problem such as a RAM fault.
          The environment variables and the contents of M: will be preserved (unless the system detects a memory corruption).
  
- 4        The user pressed Shift-Psion-Ctrl-Del to perform a hard reset. The environment variables and the contents of M: will have been cleared.

For example if the machine has lost all power, entering `reason` will return:

Reset code : 0

**REMXe "Command:REM"\$Comment (remark) in batch file****tc "Comment (remark) in f**

REM [text]

Comment (remark) in batch file. (Note the space after the command.)

The `text` characters become a comment in the batch file.

Any text on a line preceded by `REM` becomes a comment that is not executed when the batch file is run. This is useful for temporarily removing a command line, for example when debugging a batch file.

Tip: another way to temporarily remove a command line in a batch file is to precede it with a colon (:) at the start of the line. This converts the command into a label (see the `GOTO` command).

**RENxe "Command:REN"\$Rename file(s)****tc "Rename file(s) (REN)"****l 3\$xe "Rename f**

REN [system:][drive:][path]filespec filename

Changes the name of a file or files.

The command renames all files matching `filespec` - which can include wildcards.

For example, the command `ren work.* play.*` changes the names of all files called *work* in the current directory (regardless of extension) to *play*, with the extension being preserved across the rename.

As files are renamed, they are listed on the screen.

Because it is not possible to rename files from one directory to another, the command fails if any path specified with `filename` (explicitly or implicitly) differs from that of `filespec`.

It is not possible to rename a file to have the same name as a file that already exists.



**RMDIRxe "Command:RMDIR"\$Remove directorytc "Remove directory (RM**

RMDIR [system:][drive:]path [/y]

Deletes a directory, including any files in it (and subdirectories).

If the /y flag is used, no confirmation is asked for. This is intended for use in batch files.

Note that, in contrast to MS-DOS, there is no requirement to delete all the files in a directory before removing the directory.

In another difference from MS-DOS, it is perfectly possible, in the *Workabout* Command Processor, to remove the directory where the current path is. All that will happen is that subsequent commands such as `dir` may fail until such time as the current path is changed.

As files and directories are deleted, their names are listed on the screen.

The `rmdir` command does not accept a wildcard specification.

The identical command `RD` is also supported.

**SETxe "Command:SET"\$Display, set or delete environment variabletc "Dis  
"Environment variable:delete (SET)"\$xe "Display:environment  
variable (SET)"\$xe "Environment variable:display (SET)"\$**

SET [[var=[value]]][varspec]

Displays or sets the value of environment variables, or deletes them.

With no parameters, the values of all current environment variables are displayed.

If the environment variable `var` is given but without any trailing equals sign (=), the value of this environment variable is displayed.

If the equals sign (=) is given between the variable name and a value, the environment variable `var` is set to `value`.

If the equals sign is given whilst `value` is omitted, however, the environment variable `var` is deleted.

If the environment variable specification `varspec` is given, the values of all environment variables matching the specification are listed.

For example:

<code>set last=34</code>	Sets the value of <code>last</code> to the string <code>34</code> , or creates and sets it if it doesn't already exist.
--------------------------	---

<code>set last</code>	Displays the value of environment variable:
	<code>last=34</code>

<code>set last=</code>	Deletes the environment variable <code>last</code> .
------------------------	--

set \$WS\*

Displays the values of all environment variables whose names start with \$WS.

The list pauses when the screen is full.

Note that environment names and values are both case dependent. Thus the environment variables `group` and `GROUP` would be distinct.

The value (and, indeed, the name) of an environment variable may contain binary data. When displaying the content of an environment variable, non-printable byte values are displayed as `\04` or `\01`, for example. There is, however, no mechanism for *setting* binary values from the Command Processor.

**SETDEFxe "Command:SETDEF"\$Alter system settingstc "Alter system set**

SETDEF [AMnn][ABnn][DDMY | DMDY | DYMD][K0 | K1][S+ | S-][T12 | T24]

Alter system settings, (which are otherwise set by menu options). This is intended for use in batch files.

The SETDEF command on its own does nothing - it must be followed by at least one parameter.

The allowed parameters are given below.

AMnn	Auto-switchoff the machine after nn minutes of inactivity. Range 01 to 30. Values outside this range will not give an error, but the setting will not be changed.
ABnn	Auto-switchoff the backlight after nn minutes of use. Range 01 to 10. Values outside this range will not give an error, but the setting will not be changed.
DDMY	Date format DD/MM/YY. Note that the separator (/) can be changed via the menus.
DMDY	Date format MM/DD/YY.
DYMD	Date format YY/MM/DD.
Dn	Set start of week to n, where 0 is Monday, 1 is Tuesday etc.
K0	Select standard keyboard.
K1	Select special keyboard.
S+	Sound on.
S-	Sound off.

T12	Time format am-pm. Note that the separator (:) can be changed via the menus.
T24	Time format 24 hour.
TS+	Summer time on.
TS-	Summer time off.

Sound settings for beeps and key clicks, zoom and text wrap on the screen are not configurable using `SETDEF`.

The default machine settings are:

- Auto-switchoff the machine after 5 minutes of inactivity.
- Auto-switchoff the backlight after 10 minutes of use.
- Date format DD/MM/YY.
- The "start of week" default is Monday.
- Standard keyboard.
- All sound is on.
- Beeps are loud.

- Key clicks are loud.
- Time format am-pm. The separator is :.
- Summer time off.
- Zoom is on setting 3 (see table for zoom settings in the *Font sizes and zoom settings* subsection of the *Overview* at the start of this chapter).
- Wrap is off.

## **SHEETxe "Command:SHEET"\$Run the spreadsheet application Sh3tc "Ru**

[START] {SHEET|SH3} [system::][drive:][path][filename[.extension]]

Runs the spreadsheet application Sh3. Sheet is an alias for Sh3.

If preceded by *START*, Sh3 will be run asynchronously.

If no file specification is provided, and the current drive is *m:*, (and *sheet.spr* does not already exist), a dialog is presented asking:

Create "M:\SPR\SHEET.SPR"?

Otherwise, if *sheet.spr* does already exist, then it will be presented directly for editing.

If any system, drive or path is provided, but no filename, the *sheet.spr* file will be placed in, (or retrieved from), the specified location. The default directory is *\spr\*, and it is recommended that this is where all spreadsheet files are kept.

If a filename is given, a *.spr* file of that name will be created (as above) and/or presented for editing depending on whether it currently exists or not.

An extension to the filename may be provided that differs from *.spr*, but this is not recommended, since it would make identification of the file type difficult.

### **Note**

Help on using the Spreadsheet application is available by pressing Shift-Esc.

For further information on using the Spreadsheet application, see the *Workabout User Guide*.

## **SHIFTxe "Command:SHIFT"\$Shift batch file parameters**<sub>tc</sub> "Shift batch file parameters"

SHIFT procname

Moves the replaceable parameters of a batch file one position to the left. Thus %1 becomes %0, %9 becomes %8 etc.

The command SHIFT alters the values of the replaceable parameters %0 to %9 inclusive by copying each parameter into the one prior to it. Thus the value of %1 is copied to %0, the value of %2 is copied to %1, and so on. This can be used to help write a batch file that does the same operation on an unspecified number of parameters.

The SHIFT command can also be used to construct a batch file that will accept more than 9 parameters. If more than 9 parameters are specified on the command line, those that appear after the ninth (%9) can be moved one at a time into %9 by repeated use of SHIFT.

There is no reverse SHIFT command. Once the SHIFT command has been carried out, the original parameter (%0) that existed before the shift is not recoverable. The %0 parameter starts out containing the name of the batch file, including the full path.

The following batch file, PSIDEL.BTF, shows how to use the SHIFT command with any number of parameters. It deletes a list of files from a specific directory. The parameters are the directory name followed by any number of filenames.

```
ECHO off
REM PSIDEL.BTF deletes any number of files from a given directory.
REM The command uses the following syntax: psidel dir file1 file2 ...
SET fromdir=%1
:getfile
SHIFT
IF "%1"==" " GOTO end
DEL %fromdir%\%1 /y
GOTO getfile
:end
SET fromdir=
ECHO All done
```

## **STARTxe "Command:START"\$Start a process asynchronously**<sub>tc</sub> "Start a process asynchronously" (START)"\$

START procname

Launch the given process and return immediately.

Starts the first process found matching the specification in *procname*, (asynchronously). See the *Introduction to the Workabout* chapter for details of the search order.

Notes:

For a process that is an alias of another built-in program (EDIT or SHEET), using START will start a process that appears to LSEG or LPROC as an instance of the base process. This does not apply to .als files.

For example, all text and program editors are aliases of the Word application. Using START EDIT will start an instance of Word. It will thus appear in an LPROC or LSEG listing as an instance of Word. Using WORD or START WORD is banned artificially.

On the other hand, `START SHEET` will start an instance of the Sh3 application. Be aware that a process started with `SH3` or `START SH3` cannot, however, be stopped with `STOP SHEET`. The user-visible name Sheet is only used to start an instance of the *sh3.img* program running.

## **STOPxe "Command:STOP"\$Stop a processtc "Stop a process (STOP)" \l 3**

`STOP procname [/y]`

Terminates the named process or processes matching the specification in `procname`.

Processes are sent a termination message.

If the `/y` flag is used, no confirmation is asked for. This is intended for use in batch files.

To stop a specified instance of a number of running tasks, all with the same name, the exact process name must be found out and used. For example: `stop job.$09` or `stop job.$14`.

For most applications, the effect of being stopped is identical to being killed, (see the `KILL` command). The application is interrupted immediately, with no chance being provided for data being saved to file or to environment variables. However, an application can make use of an operating system service (invoked by the `ON_STOP` macro described in the *OPL Library* manual ) to specify behaviour to be invoked whenever the application is to be terminated in this way.

### **Note:**

The `STOP` command is designed for use with user-developed programs during the development process. Fully developed (and built-in Psion) applications **would normally** be stopped using the options available from their menus or those of the System Screen.

This does not apply to *.als* files.

## **TIMExe "Command:TIME"\$Display timetc "Display time (TIME)" \l 3\$xe "Dis**

`TIME`

Displays the current time, in the default format, or the format specified by the most recent `SETDEF` or menu command.

The default display format is:

`hh:mm:ss[am|pm]`

For example `09:30:15am` will be displayed if the current system time is fifteen seconds after half past nine in the morning.

Use the Time and Date option on the Time menu to change the time.

Use the `SETDEF` command or the Formats option of the Time menu to change the format in which the time is displayed.

## **TYPExe "Command:TYPE"\$Display a text filetc "Display a text file (TYPE)"**

`TYPE filename`

Prints a text or batch file to the screen.

The display pauses itself automatically after each screen full of text.

**VERxe "Command:VER"\$Display software version numbers.tc "Display software version numbers (VER)"\$**

VER

Displays the (Workabout) ROM version number, the date and time that the ROM was mastered, the (EPOC) Operating System version number, the Shell version number and the Command Processor version number.

**VOLxe "Command:VOL"\$Display the disk volume label etc.tc "Display the disk volume label:disk volume, display (VOL)"\$**

VOL [system:][drive:]

Displays information about the disk volume on the current or specified drive.

This information consists of the system, drive, volume label, media type, free space in bytes and the total space in bytes. **The battery state is also given for RAM SSDsXE "Battery state:for RAM SSDs (VOL)"\$.**

To change the volume name use the LABEL command. The label on a volume may also be changed when the volume is formatted using the FORMAT command.

**WAITxe "Command:WAIT"\$Wait for a process to complete.tc "Wait for a process to complete"**

WAIT [procname]

Suspends the Command Processor until the (named) process completes.

If a process name `procname` is given then the message:

Waiting for "procname"

is displayed and the Command Processor becomes non-interactive until such time as another process completes.

If no process name is given "any process" is substituted for "`procname`" and (for example) the message:

Waiting for "any process"

is displayed.

To break out of this mode, press `psion+esc`. A Yes/No dialog is displayed, headed:

Stop waiting for "procname"

- If `N` is pressed then a notifier stating:

Waiting for "procname"

is displayed.

- If `Y` is pressed then the waiting stops.

Commonly, this command will be used inside batch files in the following general pattern:



```
...  
<launch program asynchronously>  
...  
<some processing>  
...  
wait  
...
```

**Example:**

```
START MYSERVER  
IF ERRORLEVEL QUIT  
START MY_APP  
IF NOT ERRORLEVEL WAIT MY_APP  
KILL MYSERVER /Y
```

**Note:** The `LPROC` and `LSEG` commands can be used to determine what processes are running. `WAIT` will only wait for processes started by the Command Processor (unlike `STOP` and `KILL` which hit anything with a matching name). Neither the `LPROC` nor the `LSEG` command can be used to determine how a process was started.

For a process that is an alias of another built-in program (`EDIT` or `SHEET`), using `WAIT basename` will wait for a process that appears to `LSEG` or `LPROC` as an instance of the base process (`basename`). This does not apply to `.als` files.

For example, all text and program editors are aliases of the Word application. Using `WAIT EDIT` will wait for an instance of Word. Using `WAIT WORD` is banned artificially.

In contrast, `WAIT SHEET` can be used to wait for the termination of an instance of the Sh3 application that was started with `SHEET` or `START SHEET`. Be aware that a process started with `SH3` or `START SH3` cannot, however, be waited for with `WAIT SHEET`: you must use `WAIT SH3` instead.