FCX User's Guide for Unix

Table of Contents

Trademarks

FCX for Linux and UNIX

Welcome to FCX Version 7 for Linux and UNIX

New - self-expanding files!

FCX - running on thousands of computers worldwide every day!

Version 7 supports Linux, UNIX, Windows and VMS.

Introduction

FCX is a powerful and fast file space management facility which combines advanced data compression techniques with extensive file handling operations to minimize the amount of disk space required to store files. FCX creates sequential or random access container files which contain one or more files whose data have been compressed

Sequential container files, called transport files, may contain any number and mixture of files which have been compressed by FCX Files within the transport file may be expanded or listed as in a directory listing. Transport files may also be listed or expanded on another Linux or UNIX system, Windows, or OpenVMS. Transport files may be appended to but not otherwise modified.

Random access container files, called discpac files or simply discpacs, contain one or more files whose data have been compressed. Compressed files may be updated within a discpac, added to the discpac, deleted from the discpac, listed or expanded from the discpac. Both transports and discpacs may contain multiple versions of any given file, but discpacs have a maximum version limit specified.

Using this Help

The help system is complete with an index and glossary and the ability to search for any word. Throughout the help system are many references to other topics in the help system. These will appear as do any links on the Internet. Usually buttons are used for the links. These have special designs as follows;

Button Type	Button example
An operation.	append
Option	-output

In addition to navigating around with the buttons, browse sequences are provided. The arrows at the top of the Table of Contents will turn red when you enter a browse sequence. You can then use the arrows to view the next and previous pages in the sequence. For example, when you select an operation from the Table of Contents, you have entered a browse sequence for the operations of that product. For each operation, a browse sequence is provided to take you through all the available options for that operation.

The search works similar to Internet searches except you are only searching in this help system. The index is extensive. You can select a topic or type a word or part of a word to view more topics. The 'refresh' or 'Sync TOC' button at the top of the Table of Contents will show you where the current topic is. The red X will 'hide' the Table of Contents. To bring it back, simply click one of he buttons in the help header.

Command Line Help

Command line help is provided..

```
$ fcx ?
```

The command line help is intended as a quick reference to the command syntax and the many qualifiers available. For an experienced user, it may well suffice in place of this help system.

FCX Files

Discpac Files

FCX generates random access FCX files, or discpac files, to provide for true online archival of files. A 'directory' or index of the compressed files is maintained within the discpac file which provides for random access of each compressed file. This allows for faster retrieval of any given file. Multiple versions of files may be maintained within the discpac; these are indicated on a directory listing of the files as primary files and backup files. The total number of versions which may be kept for any given file is specified when the discpac file is created. After that, any update or refresh of the discpac will automatically delete the oldest version of a file when the maximum number is stored. Backup versions of files may be accessed using relative version numbers, i.e., the primary file is relative version n, the most recent backup file is relative number n -1, etc.

Each update of the discpac file (using either update or refresh) may specify a key (identiification string) for the set of files which is being compressed. The key may be a date, a version number or some string which uniquely identifies the set of files being added to the discpac. Files may then be accessed (listed, retrieved or removed) using this key.

Compressed files may be inserted into a discpac or extracted from a discpac. No compression or expansion of data is required. Files contained within a discpac may be extracted for transferring to another site and then inserted into a discpac at the receiving site.

Discpac files may also be transferred to another system (Linux, UNIX, Windows, VMS) and expanded using FCX for that system.

Levels of Discpac Files

Every discpac file has a structure level associated with it. This tells the software how to interpret the data contained in the discpac. FCXdiscpac Version 7 generates Level X files. Level R was the initial level. The about operation will display the current structure level.

This level is vitally important to discpac files since they are updated in place, i.e., a new copy of the discpac is not generated. You cannot write to a Level S library with Level T software without upgrading the discpac structure. For most operations this is automatic; you will see a message indicating it is being done.

Transport Files

Sequential access transport files are generated using the compress operation. Files are compressed and stored sequentially in the transport, hence, these transports may reside on any media, e.g., magnetic tape. Since these files are written sequentially, there is no 'wasted' space in them; they are usually smaller than comparable random access discpac files. However, these files may only be read sequentially and may not be modified.

If you are compressing files to be shipped to another site, either over a network or via removable media, this is the best type of file to generate. Compressed files may be selectively expanded as needed.

Transport files may also be transferred to another system (Linux, UNIX, Windows, VMS) and expanded using FCX for that system.

Levels of Transport Files

Every FCX file has a structure level associated with it. This tells the software how to interpret the data contained in the the FCX file.   Version 7 generates Level X. Several previous levels have existed and all of them are still expandable. The about operation will display the current structure level.

This level is important to some of the new features. If you have older transport files, a list -summ will tell you the structure of those files.

- self-expanding files require Level M or later
- ODS-5 files are only supported by Level M or later

Any log files which are generated also contain the structure level.

FCX Command Syntax

FCX Command Syntax

The FCX command requires the verb fcx, an operation, zero, one or two parameters and options.

Format:

```
(1) $ fcx operation [-option-1..-option-n]

(2) $ fcx operation p_file  [-option-1..-option-n]
```

The operation tells the FCX program which function it is to perform.

Transport Operations

• append	compresses one or more files and appends them to an existing transport file. (Format 1)
• compress	compresses one or more files into a single transport file. (Format 1)
• expand	expands the contents of one or more transport files. (Format 2)
• list	displays the contents of one or more transport files. (Format 2)

Discpac Operations

• create	compresses one or more files and creates a discpac file. (Format 2)
• directory	displays a directory of a discpac file. (Format 2)
• extract	extracts compressed files from a discpac and creates a transport. (Format 2)
• insert	inserts compressed files contained in a transport into a discpac. (Format 2)

- **refresh** refreshes files in an existing discpac (newer versions of the files are added to the discpac). (Format 2)
- **remove** removes selected files from a discpac. (Format 2)
- **retrieve** retrieves and expands files contained in a discpac. (Format 2)
- **update** compresses one or more files and adds them to an existing discpac. (Format 2)

Parameters

The parameter p_file (Format 2) identifies the discpac file required for discpac operations or it identifies the transport file required for transport read operations.

Sticky defaults

Options are used for specifying file paths used by FCX. The option -input is required for many operations; it identifies a file or list of files to be operated upon. More than one file specification may be provided, separated by commas. All parts of the second and succeeding file specifications may be omitted. Defaults will be taken from the preceding file specification. This is referred to as sticky defaults. For example, the following two lists of files specify the same set of files:

```
/test/*.doc,*.pdf

/test/*.doc,/test/*.pdf
```

The first example takes advantage of the support for sticky defaults, i.e., all the files will be taken from the directory /test. Since the second set of files (*.pdf) are to be taken from the same place, the path need not be specified.

Options

Each operation may have options which further identify the user's requirements to the program. Options have the form

```
-option[=value]
```

The dash (-) character is required in all cases. Not all options take a value. These are simply specified as -option.

Options use the short form -option. The long form --option is not used even though the options may use more than one character. Options may not be concatenated as -ol for -o -l. A space must precede each option.

All operations and options may be abbreviated; the number of required characters is the minimum to make the specified operation or option unique.

Getting Started with FCX

Getting Started with FCX

Using FCX is as simple as copying a file. There are many options available for selection of input files, redirecting output files and requesting messages during the operation. The options are discussed in detail in the Options Section.

This section is intended as a tutorial to get you started and illustrate how FCX may be used effectively. Not all options or variations of options will be discussed here. Throughout this section, operations and options are abbreviated in the examples as one might do when actually entering commands.

FCX does not produce any status messages unless it is requested to do so. When first using FCX, it is recommended that you use -log with all operations. Messages will be displayed indicating which file is being processed, the name of any output files and some statistical information regarding the process. This option is defaulted ON if you are using an evaluation kit.

discpac | transport

Verifying FCX is Installed and Available

The following command will tell you whether fcx is available to you, what version is installed, and when your evaluation kit will expire. If you are not running an evaluation kit, no date will be given.

```
$ fcx about
```

If you have any problems with this, see your System Manager.

UsingTransport Files

Using Transport Files

Creating transport files is as simple as copying a file. There are many options available for selection of input files, redirecting output files and requesting messages during the operation.

Transport files are generated using the compress operation. Files are compressed and stored sequentially in the transport, hence, these transports may reside on any media, e.g., magnetic tape. These files may be stored for later use or shipped to another site, hence their name.

Compressing Files

The command to compress a single file is very simple:

```
$ fcx compress -input=myfile.dat
```

This command will compress the file myfile.dat in your current working directory and create a transport file called myfile.fcx also in your current directory. No other output will be generated. All options and operations may be abbreviated.

If you just want to compress several files and don't need special selection or exclusion options, the following command will compress your files, let you monitor what is happening, and give you statistics at the end. The transport file will have the name myfiles.fcx.

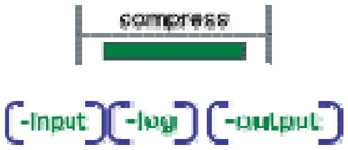
```
$ fcx comp -log -i=*.dat -out=myfiles
```

If the -out (output) option is not specified then the transport file will have the same name as the first file compressed, but with an extension of ".fcx".

-log displays messages about the progress of the compress operation. -log also lets you send the messages to a logfile. If you do not use -log you won't see any messages until all files have been compressed.

The input file list may consist of any number of files. In the example above, the input file list was simply *.dat. It could easily be expanded to be several sets of files indicated by wild cards and separated by commas.

```
$ fcx comp -i=*.com,*.lis,*.dat -out=myfiles
```



Expanding Files

To expand myfiles.fcx simply enter the following command:

```
$ fcx expand myfiles
```

The compressed files contained in myfiles.fcx will be expanded in the current working directory. If this directory is the same as the one from which the transport file was created, i.e., files of the same name and extension exist, error messages will be generated. Expanded files retain the same file name and extension of the compressed files. To override this condition, one may simply add -rep (replace) and the existing files will be deleted and replaced by the expanded ones.

The following command expands all files contained within myfiles.fcx into the directory newdir, gives a message as each file is expanded, and gives statistics at the end.

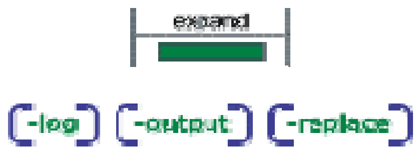
```
$ fcx exp myfiles -out=/newdir -log
```

To rename all the files in the transport file, simply add a new extension to the -out (output) file spec.

```
$ fcx exp myfiles -out=/newdir/*.new
```

In this case, all the files would be expanded into the directory /newdir and all would have the extension .new. The original file names would be kept intact.

-out (output) is necessary if you want the expanded files to reside anywhere other than the current working directory.



Listing Compressed Files

Listing the contents of a transport file is accomplished by the following command:

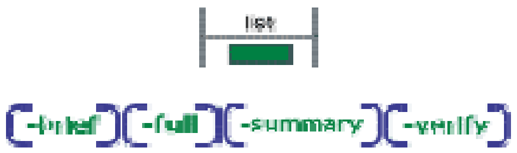
```
$ fcx list myfiles
```

This command will list the file paths of each of the compressed files. This is a brief (-brief) list by default. Use -full if you also want to see file characteristics for each file. -summary may be used to display header information about the transport file only but not each individual file name.

You can see which files are contained within a transport file and verify that the files can be expanded properly using the following command.

```
$ fcx list myfiles -v
```

-v (verify) causes the compressed files to be expanded in memory with CRC checking. If the computed CRC does not match the CRC of the original file, an error message will be output.



Compressing Multiple Files Together

Multiple files can be compressed together. Simply list all the files you want (separated by commas) using the -input switch. You may also use wildcards.

```
$ fcx comp -i=*.com,*.dat -transport=comdat
```

This will compress all the .com and .dat files in the current directory. The resultant transport file will be called comdat.fcx. It will reside in the current directory. If you want the file to be put somewhere else, simply give a directory name such as

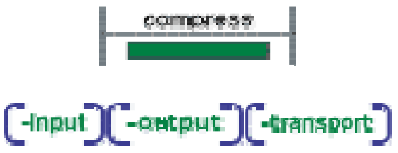
```
-transport=../otherdir/comdat.
```

The -transport option specifies the name of the resultant transport file (default file extension is fcx). If you do not specify a name, the transport file will have the same name as the first input file with extension .fcx. This may be confusing so it is a good idea to specify the name of the transport file when compressing multiple files together.

You can compress any number of files together. When specifying the list of files, the file specification parts will 'stick', i.e., if you specify a directory name, you need only specify it once if you want all the files to come from the same directory.

```
$ fcx comp -input=/mydir/*.com,*.dat
```

This command will compress the .dat files from the directory /mydir regardless of the current default.



Directories and Directory Trees

Compressing a Directory Tree

Compressing a directory is a simple matter. If -out is not specified, the transport file will have the same name as the first file compressed but with an extension of .fcx.

```
$ fcx com -i=*.* /out=mydir -log
```

Compressing a directory tree is just as simple, but you need to add the -subdir switch.

```
$ fcx com -i=*.* -sub -out=mydir  (start = current dir)

$ fcx com -i=/mytopdir/*.* -sub  (start = mytopdir)

$ fcx com -i=/*.* -sub -out=all  (start = root dir)
```

Expanding a Directory Tree

Expanding a directory is also simple. The use of -out specifies which directory the files are to be expanded into. If it is omitted, the files will be expanded into the current working directory.

```
$ fcx exp mydir -out=/mynewdir -log
```

Expanding a directory tree just requires using the option -subdir. If -subdir is omitted, all the files will go into the current working directory.

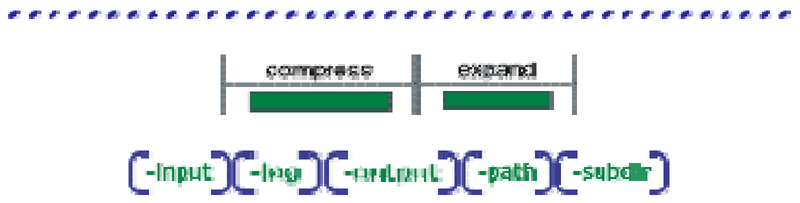
The following command will expand the files and retain the tree structure. If the original input tree contained directories that the tree structure specified by /mynewdir does not contain, FCX will create the necessary directories to preserve the tree structure.

```
$ fcx exp mydirtree -out=/mynewdirtree -subdir
```

To recreate the directory exactly as it was compressed, use the following command:

```
$ fcx exp mydirtree -log -subdir -out=/
```

This command starts the directory from the root directory of the device.



Selecting and Excluding Files

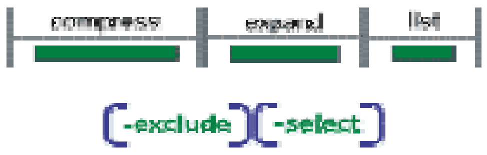
Files can be explicitly excluded from compression, expansion, or listing using the -exclude option as follows.

```
$ fcx compress -input=*.dat -exclude=f1.dat,f2.dat -out=mostdats

$ fcx exp mostdats -exclude=f3.dat
```

Files can also be explicitly selected for expansion or listing. This is most useful if one only wishes to extract a single file or a small set of files from a transport file.

```
$ fcx expand mostdats -select=f6.dat,f7.dat

$ fcx list mostdats -verify -sel=f8.dat
```

When both -select and -exclude are used, -select is applied first.



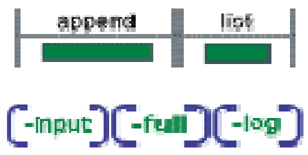
Appending Files

The syntax for appending a file is slightly different than for other transport operations. It requires both an input-file-spec and a transport-file-spec (or output-file-spec). However, it is just as simple to do and is much like the syntax used for discpac files. The command to append a single file to an already existing transport file:

```
$ fcx append myfcx -input=myfile.dat
```

This command will compress the file myfile.dat in your current working directory and append it to the transport file called myfcx.fcx also in your current directory. No other output will be generated.

The transport file which has new files appended to it looks just like any other transport file, with the exception that the files contained within it may have been compressed on different systems. To see this, a list -full is necessary. A brief list only lists the size and date of each file. For other information about individual files, a list -full is necessary.

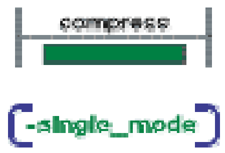


Creating Separate Transport files

It may be desirable, for a number of reasons, to create a separate transport file for each input file. File transfers with unreliable links which tend to break down are one reason. If each transport file is transmitted separately and the link breaks, only the one in progress needs be retransmitted. fcx supports this with the option -single_mode. Simply attach it to a normal fcx command.

```
$ fcx compress -input=*.dat -output=*.zzz -single_mode
```

This command will result in all the files in the current default directory with extension .dat being compressed. A single transport file will be generated for each file compressed; the transport files will have the same file names as the original files but will have an extension .zzz.



Self-Expanding transport files

Creating a self-expanding transport file is as simple as adding -self_expand to the compress command line.

```
$ fcx compress -input=*.h -transport=self_h -self_expand
```

will compress all the .h files in the current working directory and create a self-expanding file called self_h.fcxzz where zz indicates the system you are currently running on. This differentiates self-expanding files from other fcx files.

To run the file, simply type

```
$ ./self_h.fcxxl (xl in the extension stands for Linux X86)
```

This will expand the files into the current working directory. You will be prompted for a password if one was used during generation of the file

You may also redirect the output using -output. You may list the file using list or list -verify. The syntax is slightly different than FCX. You simply replace fcx on the command line with the name of the self_expanding file.

```
$ ./self_h.fcxxl list -verify

$ ./self_h.fcxxl expand -out=destination
```

You have available all the normal expand qualifiers. For list, you have -brief, -summary, and -verify.

Using Discpac Files

Using Discpac Files

Random access FCX files (discpac files) provide a mechanism to keep several versions of files compressed together in a discpac. A discpac is a 'disk within a disk'. Files within the discpac may be tagged with keys which identify a group of files.

Files within a discpac are either primary files or backup files. Primary files are the latest version of each file; these are the set of files which are listed in the directory by default. Retrieving a file simply by file name will retrieve only the primary file.

Backup files are older versions of primary files. They may have different keys than the primary files. The maximum number of backup files which may reside in the discpac for any given primary file is determined when the discpac is created. Adding files of the same name, after the maximum number of backup files has been reached, results in the oldest backup file being deleted from the discpac. This prevents the discpac from growing too large with unwanted and unnoticed files.

Creating a Discpac File

Creating a discpac file is very simple. Most of the options available may simply be defaulted:

```
$ fcx create mylib -input=*.dat
```

This command will compress the files with file type .dat in your current working directory and create a discpac file called mylib.dpx also in your current directory. No other output will be generated. If you would like to see statistics indicating how much compression was achieved, you would simply add -log to the command line. All options and operations may be abbreviated.

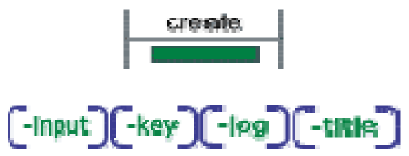
If you would like your discpac to have a title when a directory is obtained simply add -title=string where string is the set of characters for the title. If you want spaces, you will need to enclose the string in quotes.

The default maximum version count is 10. If you would like a different number add -max_versions=n to the command line. This is the number of versions (backup files plus one primary file) allowed for each file.

If you would like this particular set of files (*.dat) to have a special identification string simply add -key=string to the command line. The following command will create your discpac with a title and a key for each file.

```
$ fcx create mylib -title=dat_files -input=*.dat -key=VER_1
```

If you do not use -log you will not see any messages until all files have been compressed.



Retrieving a Compressed File

To retrieve and expand one or more files from a discpac file simply enter the following command:

```
$ fcx retrieve mylib sample.dat
```

The compressed file sample.dat (primary file only) contained in mylib.dpx will be expanded in the current default directory.

If this directory is the same as the one in which the original files resided, i.e., files of the same name and type already exist, error messages will be generated. To override this condition, one may simply add -replace to the command line and the existing files will be replaced by the expanded ones.

The following command retrieves and expands a backup file contained within the discpac mylib into the subdirectory newdir and gives statistics at the end. Backup files may be selected using the key (-key) associated with them or a relative position number.

```
$ fcx retr mylib -select=sample.dat -out=newdir/ -log
```

If the file to be retrieved had a key of VER_1 the command could be altered as follows and achieve the same result. If the key is specified, it must exactly match that of the file when it was compressed. If you are unsure of the key, use the directory operation to see exactly what it is.

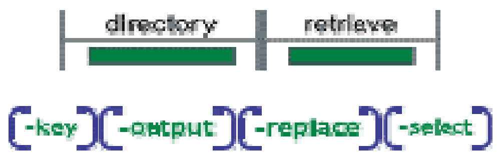
```
$ fcx retrieve mylib -select=sample.dat -key=VER_1
```

To rename all the files in the discpac file, simply add a new extension to the -out (output) file spec.

```
$ fcx retr mylib -select=*.dat -out=/newdir/*.new
```

In this case, all the files with file type .dat (primary files only) would be expanded into the directory /newdir and all would have the extension .new.

-out (output) is necessary if you want the expanded files to reside anywhere other than the current default directory.



Discpac Directories

Displaying the contents of a discpac file is accomplished by the following command:

```
$ fcx directory mylib
```

fcx will display the file specifications of each of the primary compressed files. This is a brief (-brief) list by default. Use -full if you also want to see file characteristics for each file, the key for each file and the date the file was inserted into the discpac. -summary may be used to display header information but not each individual file name.

If you would like the backup files listed also, simply add -all to the command line. If you are obtaining a brief directory (this is the default) and would like the file key listed also, simply add -key to the command line. If you specify -key=string, only those files whose key matches the specified string will be listed.

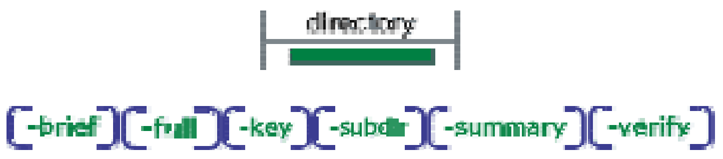
Similar to list with -select for transport files, directory will also process -select which specifies which files are to listed in the directory. The following command will list only those compressed files which have file type .dat.

```
$ fcx dir mylib -select=*.dat -all
```

You can see which files are contained within a discpac file and verify that the files can be expanded properly using the following command.

```
$ fcx dir mylib -verify
```

-verify causes the compressed files to be expanded in memory with CRC checking. If the computed CRC does not match the CRC of the original file, an error message will be output.



Updating a Discpac

The update operation provides for adding files to an existing discpac. This may be accomplished simply with the following command:

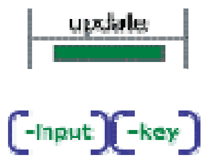
```
$ fcx update mylib -input=*.com
```

This command will add the files with file type .com from the current working directory to the discpac file mylib.dpx also in the current working directory. These .com files may be tagged with a key by using the -key=string option.

```
$ fcx update mylib -input=*.com -key=version_2
```

If the input-file-spec had also included .dat files, these files would only have been added if they did not already exist in the discpac (FCX checks the revision dates of the files).

If the revision date of the file in the current working directory is later than that of the file with the same name in the discpac, the new file is added as a primary file and the existing primary file is moved to be the latest backup file with relative version or position number of ~n where n is one higher than the latest backup file. The backup file with relative position number ~2 would be moved to relative position number ~1 and so forth. If the maximum number of versions has been reached, the oldest backup file will be automatically removed from the discpac.



Refreshing a Discpac

The refresh operation is quite similar to the update operation except that it takes as a default input-file-list the contents of the discpac. That is, the only files added to the discpac are later versions of files which already exist in the discpac. The discpac is simply kept up to date or 'refreshed'. The command for doing this is simple:

```
$ fcx refresh mylib
```

There are very few options for the refresh command. An input-file-spec may be provided which limits the files in the discpac to be refreshed.

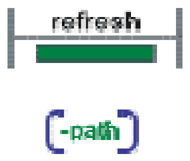
```
$ fcx refresh mylib -input=*.dat
```

This command specifies that only the files in the discpac which have file type .dat be refreshed.

As each file is refreshed, the file name is taken from the discpac and compared with the same file name in the original directory. If it is desired to refresh the discpac from the current working directory, use the option -nopath.

```
$ fcx refresh mylib -nopath
```

If the revision date of the file in the current working directory is later than that of the file with the same name in the discpac, the new file is added as a primary file and the existing primary file is moved to be the latest backup file with relative version or position number of ~n where n is one higher than the newest backup file. The oldest backup file will have relative version ~1 If the maximum number of versions has been reached, the oldest backup file will be automatically removed from the discpac.

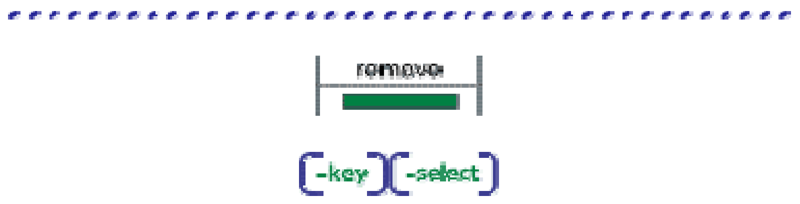


Removing Files

Removing files from a discpac file is accomplished using a command such as:

```
$ fcx remove mylib -select=sample.dat
```

This would result in all versions of the file sample.dat being removed from the discpac. If you only want one version of the file to be removed, use a relative position number such as ~2, or specify a key (-key).



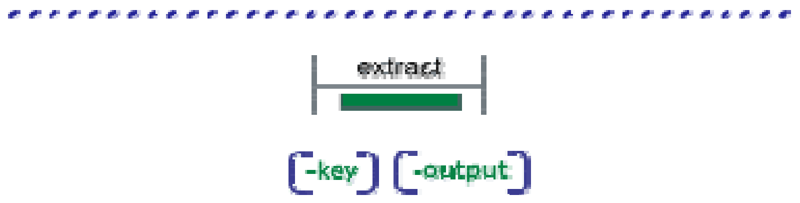
Extracting Compressed Files

If you have a large discpac and wish to transmit some of the files to a remote site, first you will want to extract them in compressed format and create a transport file for the transfer. This is accomplished using a command such as:

```
$ fcx extract mylib -select=*.* -key=Latest_Files -out=xfer
```

This command will extract all of the files in the discpac mylib.dpx which have the key string "Latest_Files". None of the files will be expanded. They will instead be put into a new transport file called xfer.fcx which is ready for transmission to another site.

Files may also be extracted using the -accessed, -created, or -modified date of each file.



Inserting Files

Complimentary to the extract operation is the insert operation. Together, these two operations provide a mix and match capability between discpac and transport files. For this example, assume you just received the transport file generated in the extract example. You now wish to insert these files into an already existing discpac on your local system. This can be accomplished with the following command:

```
$ fcx insert locallib xfer -key=New_Files
```

The newly received files, contained in the transport file xfer.fcx, will be inserted into the discpac locallib.dpx. They will have the new key string "New_Files". As with extract, the files are not expanded and then compressed, thus making these operations very fast.

You will probably want to do a directory or dir -key to see the new files in your discpac. You may now manipulate them as if you had added them with the update operation.

extract | insert

[-key]

Support Operations

about

The **about** operation is used to identify the current version of FCX.

format

```
$ fcx  about
```

description

The about operation will display the current version of FCX. If you are running an evaluation copy of FCX, it will also display the expiration date of your license..

help

The **help** operation is used to display the online help.

format

```
$ fcx  help
```

```
$ fcx  ?
```

description

The help operation will display a short synopsis of the operations and options available.

license

The **license** operation displays the FCX license file or obtains the system ID required for the license file.

format

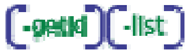
```
$ fcx  license  -options
```

description

The license operation will display the current FCX license. It may be also used to obtain the system ID required for a permanent license..

options

To obtain the system ID required for a permanent license, use -getid. To display the current license, use -list.



Transport Operations

Transport operations

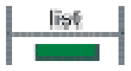
fcx generates and manipulates sequential access transport files for transfer to another site, backup, or for online archival.

operations

The append operation provides for adding newly compressed files to an already existing transport file, i.e., input files are compressed and appended to the transport file.

The compress operation is used to generate a sequential transport file containing one or more files whose data have been compressed. The original files remain intact unless deletion of these files is requested.

The expand operation is used to expand one or more compressed files contained in a transport file. The default mode is to expand all the files; this may be modified using options.



The list operation is used to generate a directory style listing of the contents of one or more transport files. Optionally, it may also be used to verify the data integrity of each compressed file in the transport. This is useful to do after a file transfer if the data is not to be expanded immediately or if there was a suspected transmission error.

description

Files are compressed and stored sequentially in the transport file, hence, these transport files may reside on any media, e.g., magnetic tape. Since these files are written sequentially, there is no 'wasted' space in them; they are usually smaller than comparable random access discpac files. However, sequential access files may only be read sequentially and may not be modified. Compressed files may be selectively expanded as needed.

Transport files may also be transferred to another system (Linux, UNIX, Windows, VMS) and expanded.

append

The **append** operation is used to compress and append files to an existing transport file.   Except for the syntax, the append operation is virtually identical to the compress operation. Depending upon the algorithm selected and the actual data in the files, compression may be greater than 90%. The original files remain intact unless deletion of these files is requested. For saving disk space, deletion of the original files would be desired. For transferring files to another site, it would not. Hence, the normal mode of FCX is simply to create a transport leaving the original files intact.

format

```
$ fcx  append  -input=input-file-spec -transport=transport-file-spec  -options
```

required options

-input=input-file-spec

The input-file-spec gives the file specification of the file(s) to be compressed. There is no default for the file name or file type; the user must supply these. The path, if not supplied, will default to the current process working directory.

More than one file specification may be provided, separated by commas.

Full wild card support is provided.

-transport=transport-file-spec

The transport-file-spec gives the file specification of an existing transport to which the compressed files will be appended. There is no default for the file name; the user must supply this. The extension will default to .fcx. The device and directory will default to the current working directory.

description

The append operation will compress the input files and append the compressed files to the already existing transport file specified by the transport-file-spec.

options

Many of the append options are used to alter selection of files from the input-file-spec. Files may be selected based on a system time parameter in the file header (see -before, -since, -accessed, -created, and -modified). Files may be excluded from selection using the -exclude and -confirm options. Other options control execution (-log, -verify).

Option Browse Sequence

-accessed | -ascii | -before | -confirm | -created | -delete | -exclude | -input | -level
-log | -modified | -since | -subdir | -transport | -verify

compress

The compress operation is used to generate a transport containing one or more files whose data have been compressed. Depending upon the algorithm selected and the actual data in the files, compression may be greater than 90%. The original files remain intact unless deletion of these files is requested. For saving disk space, deletion of the original files would be desired. For transferring files to another site, it would not. Hence, the normal mode of FCX is simply to create a transport leaving the original files intact.

format

```
$ fcx  compress -input=input-file-spec -options
```

required options

-input=input-file-spec

The input-file-spec gives the file specification of the file(s) to be compressed. There is no default for the file name or file type; the user must supply these. The path, if not supplied, will default to the current process working directory.

More than one file specification may be provided, separated by commas.

Full wild card support is provided.

description

The compression operation consists of substituting input characters with corresponding compression codes and writing the output transport file. A Cyclic Redundancy Check (CRC) is calculated for each input file and stored in the transport file. This CRC is verified by the expand operation and optionally the compress and list operations (see -verify).

Compression ratios will also vary based on the input data. Files with similar data patterns, e.g., ASCII files, will generate different compression ratios than those with more varied data patterns, e.g., binary files.

options

Command options are used to control the compress operation.

Many of the compress options are used to alter selection of files from the input-file-spec. Files may be selected based on a system time parameter in the file header (-before, -since, -accessed, -created, and -modified). Files may be excluded from selection using the -exclude and -confirm options.

Other options alter the format or destination of the resultant transport file (-output, -transport) or control the execution process (-log, -verify).

Password protection may be requested.

Separate transport files may be generated for each input file using the -single_mode option.

Self-expanding files may be generated for a Linux, UNIX, Windows, or VMS system.

Option Browse Sequence

-accessed -ascii -before -comment -confirm -created -delete -exclude
-input -level -list -log -modified -output -password -replace -self_expand
-since -single_mode -subdir -transport -verify

expand

The expand operation is used to expand one or more compressed files contained in a transport. The default mode of expand is to expand all the files; this may be modified using options.

The transport is left unchanged.

format

```
$ fcx expand transport-file-spec -options
```

parameters

transport-file-spec

The transport-file-spec gives the file specification of the transport to be expanded. There is no default for the filename; the user must supply this. The file type defaults to .fcx. The path, if not supplied, will default to the current process working directory.

description

File information describing each compressed file as well as information describing the compression codes is read from the transport. Each compressed file selected for expansion is then read and reconstructed to match the original input file. As each file is created, a CRC is calculated and compared with the CRC of the original file. If they do not match, an error message is issued.

Expanded files will match the original files both in structure and data content. Many options are available to override or alter the characteristics of the expanded files.

The expand operation may be used to expand compressed files contained in a transport which was generated on a Linux, UNIX, Windows or VMS system using fcx. The default file information may be displayed prior to expansion using the list -full operation.

Full wild card support is provided.

options

Files may be excluded from expansion using the -exclude,-select and -confirm options. Other options alter the destination of the expanded files (-output) or control the execution process (-log).

Option Browse Sequence

[-confirm][-eof][-exclude][-log][-output][-path][-raw_data][-replace][-select]
[-subdir]

list

The list operation is used to generate a directory style listing of the contents of one or more transport files. Optionally, it may also be used to verify the data integrity of each compressed file in the transport. This is useful to do after a file transfer if the data is not to be expanded immediately or if there was a suspected transmission error.

The list operation provides the display of a transport to the current stdout device. The transport is left unchanged.

format

```
$ fcx list  transport-file-spec -option
```

parameters

transport-file-spec

The transport-file-spec gives the file specification of the transport to be listed. There is no default for the filename. The file type defaults to .fcx. The path, if not supplied by the user, will default to the current process working directory.

Full wildcard support is provided.

description

File information describing each compressed file as well as information describing the compression codes is read from the transport file. Each file selected for display is then read and its original file name written to the current stdout device and optionally a log file.

options

LIST options are used to control the format of the directory listing (-brief, -display, -full, -summary). Files may be excluded from the listing using the -exclude and -select options. Other options alter the destination of the listing (-output) or verify the integrity of compressed files (-verify).

Option Browse Sequence

[-brief][-display][-exclude][-full][-output][-select][-summary][-verify]

Discpac Operations

Discpac operations

fcx may be used to generate and manipulate random access fcx files, or discpac files, which provide for online archival of files.

operations

The create operation is used to generate a random access discpac file. Only one version of any given file may be included when creating a discpac.

The directory operation is similar to the list operation for transport files. It provides a directory style listing of the contents of a discpac file. Files may be selected for inclusion in the directory listing using the key field.

The extract operation provides for extracting compressed files from a discpac (with no expansion of data) and creating a transport file to contain the compressed files.

The insert operation provides for inserting compressed files contained in a transport file into a discpac. No expansion of data is performed.

The refresh operation provides an automatic update of the discpac file. fcx looks at the original directory of each file in the discpac to see if there is a later version of the file. If there is, it is added to the discpac as the highest version (primary file). The lowest version of each file will be removed if the maximum number of versions has been reached for that file.

The remove operation is used to remove files from a discpac file. Files may be removed from a discpac by specifying the file name and file type.

The retrieve operation is used to retrieve and expand one or more compressed files contained in a discpac file. This is analogous to the EXPAND operation for transport files.

The update operation is used to add new files or new versions of files to a discpac file. Each new file is compared with the files in the discpac to see if it is a later version of a compressed file. If it is, it is added as the primary file and the existing files become backup files.

description

A 'directory' or index of the compressed files is maintained within the discpac file which provides for random access of each compressed file. This allows for faster retrieval of any given file. Multiple versions of files may be maintained within the discpac; these are indicated on a directory listing of the files as primary files and backup files. The total number of versions which may be kept for any given file is specified when the discpac file is created. When the maximum number for a given file is reached, compressing additional versions of that file will cause automatic deletion of the oldest version of the file. Backup versions of files may be accessed using version numbers; FCX adds a version number to each file starting with 1. The most recent version of a file (i.e., the primary file) will be version n..

Each update of the discpac file (using either the update or refresh operations) may specify a key (identification string) for the set of files which is being compressed. The key may be a date, a version number or some string which uniquely identifies the set of files being added to the discpac. Files may then be accessed (list, retrieve or remove) using this key.

Discpac files may also be shipped to another system (Linux, UNIX, Windows, VMS) and expanded or manipulated.

create

The create operation is used to generate a random access fcx file or discpac. A 'directory' or index of the compressed files is maintained within the discpac file which provides for random access of each compressed file. This allows for faster retrieval of any given file. Multiple versions of files may be maintained within the discpac; these are indicated on a directory listing of the files as primary files and backup files.

format

```
$ fcx  create  discpac-file-spec -input=input-file-spec -options
```

parameters

discpac-file-spec

The discpac-file-spec gives the file specification of the discpac file to be created. There is no default for the file name; the user must supply this. The file type will default to .dpx. The device and directory, if not supplied by the user, will default to the current process device and directory.

required options

-input=input-file-spec

The input-file-spec gives the file specification of the file(s) to be compressed. There is no default for the file name or file type; the user must supply these. The device and directory, if not supplied, will default to the current process device and directory.

More than one file specification may be provided, separated by commas. All parts of the second and successive file specifications may be omitted. Defaults will be taken from the preceding file specification.

Full wild card support is provided, except for versions.

description

An index or directory of the files contained within the discpac file is maintained by fcx within the discpac file itself. This provides for true random access of the compressed files. Each time a file is added to the discpac it is also added to the discpac index. When a file is removed from the discpac, its entry is removed from the index. The data portion of the file is still retained within the discpac until the next time a file is added. The newly added file will use the same file space previously occupied by the removed file.

Inherent with discpac files is the concept of primary and backup files. When a discpac is created, only primary files may be inserted into the discpac, i.e., only one version of the file may be inserted. With subsequent updates to the discpac, newer versions of the primary files may be added. As this occurs, each existing primary file is 'moved' to a backup file. (The file is not actually moved. Only its internal status is updated in the discpac index). Files are moved to backup status on a push down stack basis. This keeps the newest file as the primary file and the oldest file as the lowest version on the stack. The total number of primary and backup files allowed for any given compressed file is specified during discpac creation (See -max_versions). The default is ten. Once the limit is reached for any given file, the oldest backup file is removed from the discpac. Backup files may be accessed using version numbers where ~1 indicates the fist file inserted into the library. The most recent file is ~n.

Files within the discpac may be designated by a key. This is simply an identification string which is used to denote a set of files which were added to the discpac together, either with the create, update or refresh operations.

Discpacs, since they are accessed randomly, may only reside on a random access device. They may not be created nor updated on magnetic tape. They may however, be transferred to another fcx supported system (Linux, UNIX, Windows, or VMS) system. Files may be retrieved on these other systems in the same manner as transport files.

options

Many of the create options are used to alter selection of files from the input-file-spec. Files may be selected based on a system time parameter in the file header (see -before, -since, -accessed, -created, and -modified). Files may be excluded from selection using the -exclude and -confirm options.

Other options specify discpac control options (-key, -max_versions, -title) or control the execution process (-log, -verify). Password protection is also available (-password).

Option Browse Sequence

[-accessed] [-ascii] [-before] [-confirm] [-created] [-delete] [-exclude] [-input] [-key] [-level] [-log] [-max_versions] [-modified] [-password] [-replace] [-since] [-subdir] [-utc] [-verify]

directory

The directory operation is similar to the list operation for transport files. It provides a directory listing of the contents of a discpac to the current stdout device. Files may be selected for listing using the key (-key) field. All versions of files may be listed. The default is to list only the primary files (latest version). All versions of each file (primary files and backup files) may be displayed (see -all).

The discpac file is not modified.

format

```
$ fcx  directory discpac-file-spec  -options
```

parameters

discpac-file-spec

The discpac-file-spec gives the file specification of the discpac to be displayed. There is no default for the filename; the user must supply this. The file type defaults to .dpx. The device and directory, if not supplied by the user, will default to the current process device and directory.

description

File information describing each compressed file is read from the discpac index. Optionally, files may be verified for data integrity using CRC checking; in this case each file selected is then read and expanded in memory.

Only one discpac file may be accessed at a time.

More than one comp-file-spec may be provided, separated by commas.

options

directory options are used to control the format of the directory listing (-brief, -display, -full, -summary). Files may be selected for or excluded from the directory display using the -all, -exclude and -key options.

Other options alter the destination of the directory listing (-output) and provide for verification of the discpac (-verify).

Option Browse Sequence

[-all] [-brief] [-display] [-exclude] [-full] [-key] [-output] [-select] [-summary] [-verify]

extract

The extract operation provides for extracting compressed files from a discpac file and creating a transport file. Once the files have been extracted, transport operations apply to the new transport file. The transport may be listed and files expanded from it. The original discpac file remains intact and still contains the compressed files which were extracted. This operation eliminates the need to retrieve files (and expand the data) and then compress them into a transport for file transfers.

format

```
$ fcx  extract  discpac-file-spec -transport=transport-file-spec -options
```

parameters

discpac-file-spec

The discpac-file-spec gives the file specification of the discpac file to be accessed. There is no default for the filename; the user must supply this. The file type defaults to .dpx. The path, if not supplied by the user, will default to the current process working directory.

required options

-transport=transport-file-spec

The -transport option is required to specify the file name of the transport to be created.

description

Compressed files are extracted from a discpac in compressed format as transport files. These files may then be treated as other transports and the list and expand operations now apply as opposed to the retrieve and directory operations. These files may then be transferred to another site or even inserted into another discpac using the insert operation.

options

Files may be extracted from a discpac in the same manner as they are expanded, i.e., relative version numbers are supported. Files may be extracted by key (-key).

The normal selection options also apply (-confirm, -exclude, and -key). A comment (-comment) may be included in the transport file being created. The -transport option is required to specify the destination of the transport file.

Option Browse Sequence

[-comment] [-confirm] [-exclude] [-key] [-log] [-output] [-select] [-transport]

insert

The insert operation is the reverse of the extract operation, i.e., it inserts compressed files from a transport file into a discpac. Again, no compression or expansion of data is performed. Files may be inserted from transports created on a different system (Linux, UNIX, Windows, VMS) using FCX

format

```
$ fcx  insert  discpac-file-spec -transport= transport-file-spec -options
```

parameters

discpac-file-spec

The discpac-file-spec gives the file specification of the discpac file(s) to be accessed. There is no default for the filename. The file type defaults to .dpx. The path, if not supplied by the user, will default to the current process working directory.

required options

transport-file-spec

The transport-file-spec gives the file specification of the transport file whose compressed data are to be inserted in the discpac. There is no default for the filename. The file type defaults to .fcx. The path, if not supplied by the user, will default to the current process working directory.

description

Compressed files contained within a transport file may be added to a discpac with the insert operation. The transport file is not modified. Files are not expanded as they are inserted, only the internal structure is changed. These files may then be expanded from the discpac using the retrieve operation. This operation allows for maintaining a discpac where files are transferred from elsewhere in the form of transport files.

options

Files may be selected from the transport file using -confirm, -exclude, and -select. A key (-key) may be used to tag files in the discpac file.

Option Browse Sequence

-exclude -key -select -transport

refresh

The refresh operation is an automatic update of the discpac file. FCX looks at the original directory of each file in the discpac to see if there is a later version of the file. If there is, it is added to the discpac as the highest version (primary file). The lowest version of each file will be removed if the maximum number of versions has been reached for that file. Use of the -nopath option requests FCX to look at the current directory for later versions of files rather than the original directory.

format

```
$ fcx refresh  discpac-file-spec -options
```

parameters

discpac-file-spec

The discpac-file-spec gives the file specification of the discpac file to be updated. There is no default for the file name; the user must supply this. The file type will default to .dpx. The device and directory, if not supplied by the user, will default to the current process device and directory.

description

The device and directory of each primary file in the discpac is checked to see if a later version of the file exists. If it does, it is added to the discpac as the new primary file; the existing primary file is pushed down to a backup file. If the maximum number of versions has been reached, the oldest backup file is removed from the discpac. If it is desired to refresh a discpac from a directory other than the original, the option -nopath may be used.

options

refresh options may be used to alter selection of files from the discpac (-confirm, -key and -select).

Option Browse Sequence

[-confirm][-delete][-exclude][-key][-level][-log][-path][-select][-verify]

remove

The remove operation is used to remove files from a discpac file. Files are removed from a discpac by specifying the file name and file type. By default, all files of the same name are removed, i.e., the primary file as well as all backup files. If a particular version of the file is desired, the version number may be specified. In this case, only the specified file is removed; the remaining backup files and/or the primary file are left in tact. The key (see -key) of the file may also be used to select files.

format

```
$ fcx remove  discpac-file-spec -select=select_file_spec -options
```

parameters

discpac-file-spec

The discpac-file-spec gives the file specification of the discpac file to be updated. There is no default for the file name; the user must supply this. The file type will default to .dpx. The device and directory, if not supplied by the user, will default to the current process device and directory.

required options

-select=select-file-spec

The select-file-spec gives the file specification of the file(s) to be removed from the discpac..

description

By default, all files of the same name are removed, i.e., the primary file as well as all backup files. If a particular version of the file is desired, the version number may be specified. In this case, only the specified file is removed; the remaining backup files and/or the primary file is left in tact. The key (see -key) of the file may also be used to select files.

The file space occupied by removed files is returned to a pool of free space in the discpac index. This space is then available for addition of new files. Removal of files does not reduce the size of the discpac.

options

remove options may be used to select files from the discpac-file-spec (-exclude, -key and -select).

Option Browse Sequence

[-all][-confirm][-exclude][-key][-log][-select]

retrieve

The retrieve operation is used to retrieve and expand one or more compressed files contained in a discpac file. This is analogous to the expand operation for transport files.

Files may be retrieved from a discpac by specifying the file name and file type. If a particular version of the file is desired, the relative version number may be specified. The key (-key) of the file may also be used to select files. By default, only the primary file is retrieved.

The discpac file is not modified.

format

```
$ fcx  retrieve  discpac-file-spec  -select=select-file-spec
```

parameters

discpac-file-spec

The discpac-file-spec gives the file specification of the discpac file(s) to be accessed. There is no default for the filename. The file type defaults to .dpx. The device and directory, if not supplied by the user, will default to the current process device and directory.

required options

-select=select-file-spec

The select-file-spec gives the file specification of the compressed file(s) to be retrieved from the discpac. There is no default for the file name or file type; the user must supply these. The path, if not supplied by the user, will default to the original path. Version numbers will default to the primary file only.

More than one file specification may be provided, separated by commas.

description

File information describing each compressed file is read from the discpac file. Each compressed file selected for expansion is then read and reconstructed to match the original input file. As each file is created, a Cyclic Redundancy Check (CRC) is calculated and compared with the CRC generated for the original file. If they do not match, an error message is issued.

Retrieved files will match the original files both in structure and data content.

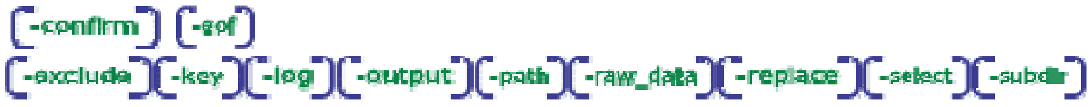
Files may be retrieved from a discpac by specifying the file name and file type. If a particular version of the file is desired, the version number may be specified. The key strings (see -key) associated with the files may also be used for selection. By default, only the primary file is retrieved.

The retrieve operation may be used to expand compressed files contained in a discpac file which was generated on a different system (Linux, UNIX, Windows, VMS).

options

Options alter the destination of the expanded files (-replace, -output) or the format of the expanded files (-eof, -raw_data).

Option Browse Sequence



update

The update operation is used to add new files or new versions of files to a discpac file. Each new file is compared with the files in the discpac to see if it is a later version of a compressed file. If it is, it is added as the primary file and the existing files become backup files. If the new file does not exist in the discpac, it is simply added as a primary file. The oldest version of each file will be removed if the maximum number of versions has been reached for that file. Use of the -nopathr option requests FCX to look at the current directory when comparing file names for later versions of files; the device and directory of the files are ignored.

format

```
$ fcx  update  discpac-file-spec  -input=input-file-spec -options
```

parameters

discpac-file-spec

The discpac-file-spec gives the file specification of the discpac file to be updated. There is no default for the file name; the user must supply this. The file type will default to .dpx. The device and directory, if not supplied by the user, will default to the current process device and directory.

required options

-input=input-file-spec

The input-file-spec gives the file specification of the file(s) to be compressed. There is no default for the file name or file type; the user must supply these. The device and directory, if not supplied by the user, will default to the current process device and directory.

More than one file specification may be provided, separated by commas. All parts of the second and successive file specifications may be omitted. Defaults will be taken from the preceding file specification.

Full wild card support is provided, with the exception of version numbers.

description

If a file of the same name and type already exists in the discpac, the new file is added as a primary file and the file already within the discpac is 'pushed down' to a backup file. Backup files already within the discpac are pushed down one level and the oldest file removed if the maximum number of versions has been reached.

The device and directory of each file to be added is checked against the device and directory of files already within the discpac. If the full pathname matches, the file is updated as described above (the new file is added as a primary file and the current primary file is pushed down to a backup file). It may be desired to update a discpac from a different device or directory. In this case, only the file name and file type are checked against what exists within the discpac. This is accomplished through the use of the -nopath option. The device and directory of each file is carried with it in the discpac.

options

Many of the update options are used to alter selection of files from the input-file-spec. Files may be selected based on a system time parameter in the file header (-before, -since, -accessed, -created, and -modified). Files may be excluded from selection using the -exclude and -confirm options.

Other options specify discpac control options (-key) or control the execution process (-log, --verify).
 Input files may be deleted using -delete.

Option Browse Sequence

[-accessed][-ascii][-before][-confirm][-created][-delete][-exclude][-input][-key]
[-level][-log][-modified][-path][-since][-subdir][-verify]

Options

Options Reference

Each option is presented with a detailed description of its function, applicable operations, and its format..

Note:

Option Browse Sequence

[-accessed][-all][-ascii]
[-before][-brief]
[-comment][-confirm][-created]
[-delete][-display]
[-eof], [-exclude]
[-full]
[-getid]
[-input]
[-key]
[-level][-list][-log]
[-max_versions][-modified]
[-output]
[-path][-password]
[-raw_data][-replace]
[-select][-self_expand][-since][-single_mode][-subdir][-summary]
[-title][-transport]
[-verify]

Appendix

VMS File Specification Syntax

A VMS file specification consists of a device specification, a directory path, a file name, a file extension and a version number. The file name and extension are separated by a period (.). The extension and version number are separated by a semi-colon (;). The directory path consists of a set of subdirectory names separated with the period (.) character and enclosed within the bracket ([] or <>) characters. The device specification consists of a device name followed by a colon (:). The complete file specification of the file TEST.DAT might be

```
DUA0:[MYDIR]TEST.DAT;1
```

Logical names and search lists may be used in file specifications. For a complete description of VMS file specifications including wild cards and sticky defaults, refer to "Guide to OpenVMS File Applications".

Directories may also be specified by relative paths to the current default directory or the master file directory (root directory). The following conventions are used by FCX:

Format	Interpretation
`[000000]`	Master File Directory or Root Directory
`[]` (or none)	Current Default Directory
`[dirname]`	Subdirectory based from Root directory
`[-]`	Parent directory of current directory
`[-.dirname]`	Subdirectory based from parent directory of current directory

Directory trees are supported by FCX. Access to directory trees is given by the following conventions:

Format	Interpretation
`[...]`	Directory tree based from current directory
`[*...]`	Directory tree based from root directory (entire disk)

`[dirname...]`	Directory tree based from directory

Windows, UNIX, and VMS File Specifications

A Windows file specification consists of a drive specifier, an optional directory path, a file name and a file extension. The file name and extension are separated by a period (.). The directory path consists of a set of subdirectory names separated with the backslash (\) character and ending with the backslash (\) character.

UNIX and LInux file paths are similar to those of Windows with the exception of the drive letter. The backslash (\) character is replaced with the forward slash (/) character.

File specifications on VMS systems are quite similar; they consist of a device name, a directory path, a file name, a file type (or extension) and a version number. The file name and file type are separated by a period (.); the file type and version number are separated by a semicolon (;). The directory path consists of a set of subdirectory names enclosed in brackets ([]) and separated by periods (.). The complete file specification may be up to 256 characters,except for ODS-5 structure disks.

The following examples illustrate the difference between Windows file names and VMS file names.

Windows Filename	UNIX/Linux Filename	VMS Filename
`TEST.DOC`	`test.doc`	`TEST.DOC;1`

Directories may also be specified by relative paths to the current default directory or the master file directory (root directory). The following conventions are used:

Windows Format	UNIX/Linux Format	VMS Format	Interpretation
`\`	`/`	`[000000]`	Master File Directory or Root Directory
.\ (or none)	`./`	`[]`	Current Default Directory
`\dirname`	`/dirname`	`[dirname]`	Subdirectory based

			from Root directory
`dirname\`	`dirname/`	`[.dirname]`	Subdirectory based from current directory
`..\`	`../`	`[-]`	Parent directory of current directory
`..\dirname\`	`../dirname/`	`[-.dirname]`	Subdirectory based from parent directory of current directory

Glossary

<

<CTRL-C>: Generated by holding the control (CTRL) key down and pressing the C key. Used to gain the attention of an enabling process which usually cancels the operation in progress. If not enabled, then it is treated as a <CTRL-Y>.

<CTRL-Y>: Generated by holding the control (CTRL) key down and pressing the Y key. Used to interrupt the current process. Usually aborts the operation in progress.

B

Backup file: A version of a file in an FCX library file which is not the most recent. A library may contain many backup versions of a file, indicated with a relative version number lower than 0.

C

Cluster size: The fundamental unit of allocation (expressed in blocks) on a volume.

CRC: Cyclic Redundancy Check. An error detection scheme in which the check character is generated by taking the remainder after dividing all the serialized bits in a block of data by a predetermined binary number.

P

Piggyback: Refers to the concatenation of two or more error messages. Quite often an FCX error message is concatenated with an RMS error message.

Positional qualifier: A qualifier which can appear anywhere on the command line, but the meaning of the qualifier depends on the position in which it is used. If the qualifier is used after a parameter, it applies only to that parameter. If it is used after the operation, the qualifier applies to all parameters.

Primary file: The most recent version of a file in an FCX library file, indicated with a relative version number of n where n is the number of backup files plus one.

R

Random access file: An FCX file which contains many compressed files which are organized randomly. These files are also called library files. Compressed files may be added to or removed from a library file.

Relative position: Refers to the position of a file relative to the most recent file in an FCX library; the most recent file is referenced as version ;0; the next most recent file is referenced as version ;-1 etc. This is also referred to as the relative version.

Relative version: Refers to the version of a file relative to the highest version; the next most recent file is version ;-1 etc. This term applies to VMS files as well as backup files within an FCX library.

Relative version 0: Refers to the highest version of a file.

RMS: Record Management Services. The part of VMS which handles files and records within files.

S

Save-set: Files created by the Backup Utility.

Sequential FCX file: An FCX file which contains many compressed files which are organized sequentially. Compressed files may not be removed from a sequential file.

Sticky default: Means that file specification components from the first file specification are applied as defaults to the next file specification component, eliminating the need, for instance, to specify the device specification for each file specification when all files are located on the same device.

T

transport: A sequential FCX file generated by FCX for OpenVMS, DiscPac for Windows, or Thoro'Pac for Windows. It has a file type of .FCX and may reside on disk or tape. Similar in format to a BACKUP saveset.

U

UIC: User Identification Code. Used to determine the access rights of users to various objects on the VMS system, including files.

V

Version 0: Refers to the highest version of a file.

Index