Component Files

General	
`R ← ⎕FAVAIL`	Checks the file system availability (returns 1 if available, 0 otherwise)
`R ← ⎕FNUMS`	Lists the tie numbers of all tied files
`R ← ⎕FNAMES`	Lists the names of all tied files
`R ← ⎕FLIB Y`	Lists the names of the component files in directory Y
File operations	
`{R} ← X ⎕FCREATE Y` *Variants can be specified*	Creates a new file with name (and, optionally, file size limit in bytes) X and file tie number (and, optionally, address size – 64) Y; a tie number of 0 allocates the next available tie number to the file
`{R} ← X ⎕FTIE Y`	Exclusively-ties the file that has name X using file tie number* Y
`{R} ← X ⎕FSTIE Y`	Share-ties the file that has name X using file tie number* Y
`{R} ← ⎕FUNTIE Y`	Unties all files that have a tie number in vector Y
`R ← X ⎕FCOPY Y` *Variants can be specified*	Copies the file that has tie number* Y to the new name X – this can be used to convert small-span files to large-span files without altering component access timestamps
`{R} ← X ⎕FERASE Y`	Erases the tied file that has name X and file tie number* Y
`{R} ← X ⎕FRENAME Y`	Renames the exclusively-tied file that has file tie number* Y to have name X
`R ← ⎕FHIST Y`	Returns the history of the file that has file tie number* Y
`R ← ⎕FSIZE Y`	Returns information on the number of the first component within the file that has file tie number* Y, the number of the next component to be appended, the current file size and the file size limit
`{R} ← {X} ⎕FRESIZE Y`	Relocates components within the file that has file tie number* Y to eliminate any redundant space between them and reduces the file size to a maximum size X – if X is not specified then the maximum possible size is allocated to the file
`R ← X ⎕FPROPS Y`	Reports and sets the properties of the file that has file tie number* Y according to the identifiers specified in X
`R ← ⎕FCHK Y` *Variants can be specified*	Checks untied file Y – variants can be specified to indicate the action to take if this validation fails, for example `R ← ⎕FCHK ⍠ 1 ⊢ Y` attempts to repair file Y if a problem is found
Component operations	
`{R} ← X ⎕FAPPEND Y`	Appends array X as a component to the file that has tie number* Y
`{R} ← X ⎕FREPLACE Y`	Replaces a component in the file identified by file tie number and component number* Y with X
`R ← ⎕FREAD Y`	Reads the content of a component in the file identified by file tie number and component number* Y
`R ← ⎕FRDCI Y`	Returns information on the size of the component file that has file tie number* Y, the user number of the user who last updated it and the time since it was last updated in sixtieths of a second since 1 January 1970
`{R} ← ⎕FDROP Y`	Drops a block of components from the file as identified by Y – this comprises the file tie number* and the number of components to be dropped (a positive number indicates they are to be dropped from the beginning of the file, a negative number indicates they are to be dropped from the end of the file)
Manipulating access to a file	
`{R} ← X ⎕FSTAC Y`	Sets the access permissions of the file with file tie number* Y according to access matrix X
`R ← ⎕FRDAC Y`	Returns the access matrix for the file that has file tie number* Y
`{R} ← {X} ⎕FHOLD Y`	Holds the files that have file tie numbers* Y; X is an optional timeout (in ms)

* indicates that Y can, optionally, also include a passnumber

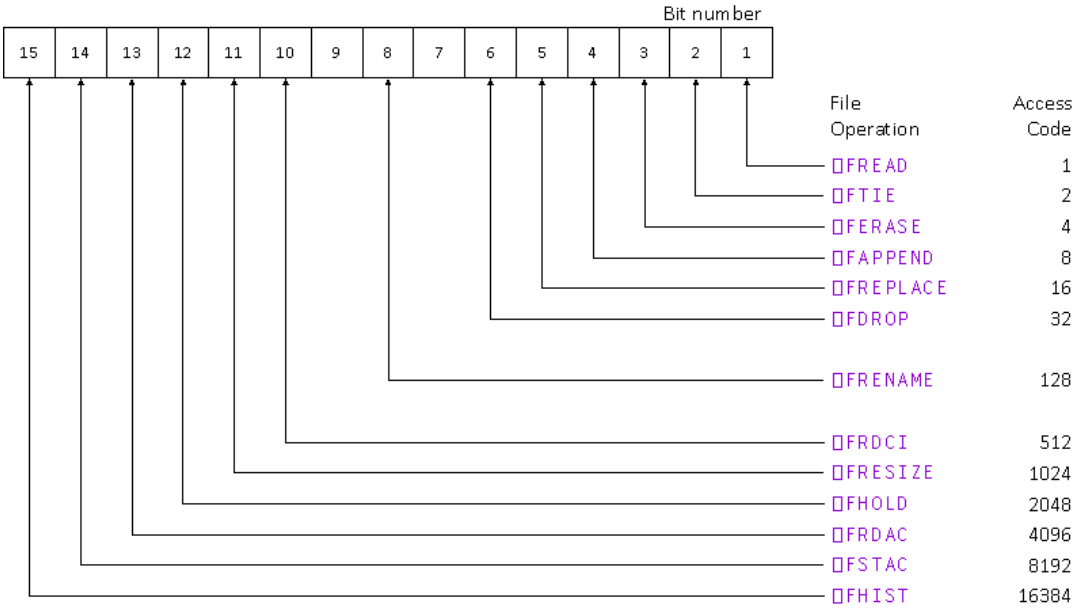
If no file extension is specified with a filename, then an extension of **.dws** is assumed (Microsoft Windows version only).
See the Dyalog documentation set for full details of these system functions, including variant options

Access Control

Dyalog's access matrix is an integer matrix with 3 columns and any number of rows.

- column 1 contains user numbers (as defined by the APLNID environment variable)
- column 2 contains an encoding of permitted file operations
- column 3 contains passnumbers

Each row specifies the file operations that can be performed by that row's user number/passnumber combination. The pertinent file operations and their associated access codes are shown in the following integer representation of a Boolean mask (where each bit in the mask indicates whether or not a particular file operation is permitted):



To determine the appropriate access code for a user, sum the access codes for that user's permitted file operations. A value of ¯1 (all bits set) permits all operations; this means that an alternative way of determining the appropriate access code for a user is to subtract the access codes of any file operations that are forbidden to that user from -1. Note that:

- Any non-zero permission code allows ⎕FSTIE and ⎕FSIZE
- ⎕FCREATE, ⎕FUNTIE, ⎕FLIB, ⎕FNAMES and ⎕FNUMS are not subject to access control
- Passnumbers can be used to establish different levels of access for the same user

Component File Properties (⎕FPROPS)

Identifier	Property	Valid Values	Default
S	File Size (read-only)	32 = small-span component files (maximum file size < 4 GB)	64
		64 = large-span component files	
E	Endianness (read-only)	0 = little-endian	depends on computer architecture
		1 = big-endian	
U	Unicode	0 = characters are written as type 82 arrays	1 for Unicode edition and large-span file, 0 otherwise
		1 = characters can be written as Unicode arrays	
J	Journaling	0 = disable journaling	1 (can be changed using the APL_FCREATE_PROPS_J environment variable)
		1 = enable *APL crash proof* journaling	
		2 = enable *System crash proof* journaling; repair needed on recovery	
		3 = enable full *System crash proof* journaling	
C	Checksum	0 = disable checksum	1 (can be changed using the APL_FCREATE_PROPS_C environment variable)
		1 = enable checksum	