uc4500 (view source)

Defined in: uf_cfi.h

Overview

Open a file for binary buffered I/O. The open mode
is used to indicate how the file is to be accessed.
The file format is used both to specify a default extension to use and
for the write modes, the type to be stored in the file's header. For the
read mode, a format of zero may be specified indicating no default
extension.

A native operating system file may be opened by using the
appropriate file type. For example, a format of 4 opens a text file in
the native file system.

A successful open returns a simple integer channel number. This
channel number is to be passed to the routines for reading, writing,
skipping, and closing the file.

uc4510 (read characters), uc4511 (read integers), uc4512 (read reals),
and uc4513 (read bytes) are used to read primitive data types from a
file open with uc4500. Combinations of these types may be used to
create more complex data types. The I/O channel number returned by uc4500
is passed to these other routines.

uc4520 (write characters), uc4521 (write integers), uc4522 (write
reals), and uc4523 (write bytes) are used to write primitive data types
to a file opened with uc4500. Combinations of these calls are used to
create more complex types. The I/O channel number returned by uc4500
is passed to these other routines.

uc4530 (skip characters), uc4531 (skip integers), uc4532 (skip reals),
and uc4533 (skip bytes) are used to bypass primitive data types in a
file opened with uc4500. A combination of these may be used to skip
over more complex types. Skips may be backward by using negative
skip counts.

For characters, the count will be rounded up to the next integer
boundary (e.g. a write/read/skip of 5 will actually write/read/skip 8
characters).

Return

Return code:
<0 = error
>= 0 = i/o channel number

Environment

Internal and External

See Also

uc4510
uc4511
uc4512
uc4513
uc4520
uc4521
uc4522
uc4523
For description of modes see table
For description of file formats see table

Required License(s)

gateway

```
int uc4500
(
    const char * fspec,
    int omode,
    int ftype
)
```

const char *	**fspec**	Input	File To Open
int	**omode**	Input	Open Mode 1 = Read 2 = Write 3 = Write, Replace 5 = Update 7 = Scratch
int	**ftype**	Input	File Format

---

uc4504 (view source)

Defined in: uf_cfi.h

Overview

Open a text file for I/O. Only the routines uc4514,
uc4524, and uc4525 for reading and writing lines may be used on text
file. See the description of uc4500 for a description of the parameters.
For omode = 4, the file must exist and will be opened for write. The
file pointer will be initially positioned at the end of file.
For omode = 6, a printer is opened. If the filespec is blank, the
default printer will be opened.

Return

Return code:
< 0 = error
>= 0 = I/O Channel Number

Environment

Internal and External

See Also

uc4514a
uc4524
uc4525
For description of file types see table

Required License(s)

gateway

```
int uc4504
(
    const char * fspec,
    int omode,
    int ftype
```

)

const char *	**fspec**	Input	File To Open
int	**omode**	Input	Open Mode 1 = Read 2 = Write 3 = Write, Replace 4 = Append 6 = Printer
int	**ftype**	Input	File Type

---

uc4506 (view source)

Defined in: uf_cfi.h

Overview

Open a binary file for block I/O. Only the routines uc4516 and uc4526 for reading and writing blocks may be used on a file opened with uc4506. See the description of uc4500 for a description of the parameters.

Return

Return code:
> 0 = error
<= 0 = I/O Channel Number

Environment

Internal and External

See Also

uc4516
uc4526

For description of file types see table

Required License(s)

gateway

```
int uc4506
(
    const char * fspec,
    int omode,
    int ftype
)
```

const char *	**fspec**	Input	File To Open
int	**omode**	Input	Open Mode 1 = Read 2 = Write 3 = Write, Replace 5 = Update 7 = Scratch

int	**ftype**	Input	File Type

---

uc4507 (view source)

Defined in: uf_cfi.h

Overview
Open a file for record oriented I/O. Only the
routines uc4517 and uc4527 may be used to read and write records.
Record I/O is generally only supported for native files.

Return
Return code:
< 0 = error
>= 0 = I/O Channel Number

Environment
Internal and External

See Also
uc4517
uc4527
For description of file types see table

Required License(s)
gateway

```
int uc4507
(
    const char * fspec,
    int omode,
    int ftype,
    int rectype
)
```

const char *	**fspec**	Input	File Specification
int	**omode**	Input	Open Mode 1 = Read 2 = Write 3 = Replace 5 = Update 7 = Scratch
int	**ftype**	Input	File Type
int	**rectype**	Input	Record Format = 0 = Variable Length Records > 0 = Fixed Record Length

---

uc4508 (view source)

Defined in: uf_cfi.h

Overview

Open a directory for reading. Directory entries are read
using uc4518 with the fields picked out using uc4600 to uc4609.
Sub-directories may be opened using uc4509.

The dmode bit-mask is used to indicate the intention of the
application in using the directory. Bits zero through seven indicate
which fields from the file header to read and bits ten through thirteen
specify special directory options.

You can specify which bits to set by using the left shift operator in
conjuction with the bit wise or. For example, to set bits 11 and 13, you
would use the following declaration:
int dmode = (1 << 11) | (1 << 13);

A wildcard template may be used and is specified when the directory
is opened. It may be passed separate from the directory filespec or as
part of the directory filespec. For example:

dmode(bit 11) = 1, fspec = "disk/manager", wcard = ".prt" will
return out all parts in the manager's directory.

If neither dmode(bit 11) or dmode(bit 12) is set then all files will be
returned. dmode(bit 11) and dmode(bit 12) should not be
set at the same time.

If dmode(bit 13) is set then the caller is only interested in the names
of the files and none of the header attributes. If this is set then none
of the parameter dmode(bit 0) through dmode(bit 7) should be set.

If dmode(bit 10) is set then the caller is intending to open
sub-directories. If it is not set then uc4509 should not be called.

Return

Return code:
< 0 = error
>= 0 = I/O Channel Number

Environment

Internal and External

See Also

uc4518
uc4600
uc4609
uc4509
For description of file types see table

Required License(s)

gateway

```
int uc4508
(
    const char * fspec,
    int dmode,
    int ftype,
    const char * wcard
)
```

const char *	**fspec**	Input	Directory Specification
int	**dmode**	Input	Bit-mask Of Open Options bit 0 = Read Owner Field 0 = No 1 = Yes bit 1 = Read Protection Classfield 0 = No 1 = Yes bit 2 = Read Status Field 0 = No 1 = Yes bit 3 = Read Length Field 0 = No 1 = Yes bit 4 = Read Dates 0 = No 1 = Yes bit 5 = Read Description Field 0 = No 1 = Yes bit 6 = Read Customer Area Field 0 = No 1 = Yes bit 7 = Read Machine Field 0 = No 1 = Yes bit 8-9 = Reserved bit 10 = Enable Sub-trees 0 = No 1 = Yes bit 11 = Template Given 0 = No 1 = Yes, Template In wcard bit 12 = Directory Contains Wildcards 0 = No 1 = Yes bit 13 = Read Filenames Only 0 = No 1 = Yes bit 14-15 = Reserved
int	**ftype**	Input	This argument is no longer used.
const char *	**wcard**	Input	Wildcard Template

uc4509 (view source)

Defined in: uf_cfi.h

Overview

Open a subdirectory of the currently open directory.
The previous directory entry read by uc4518 must have been of type
directory (format 100-112) otherwise the error 'not a directory'
is returned. Subdirectories are closed using uc4549.

Environment

Internal and External

See Also

uc4518
uc4549

Required License(s)

gateway

```
int uc4509
(
    void
)
```

---

uc4510 (view source)

Defined in: uf_cfi.h

Overview

Read characters from a file opened with uc4500.

For characters, the count will be rounded up to the next integer boundary (e.g. a read of 5 will actually read 8 characters).
Integer variables are used for storing the characters in byte format.

Environment

Internal and External

See Also

uc4500

Required License(s)

gateway

```
int uc4510
(
    int chan,
    int len,
    char cbuf [ UF_MAXWORD ]
)
```

int	**chan**	Input	I/O Channel Number returned from uc4500.
int	**len**	Input	Number Of Characters To Read
char	**cbuf [UF_MAXWORD]**	Output	Array To Read Characters Into

---

uc4511 (view source)

Defined in: uf_cfi.h

Overview

Read integers from a file opened with uc4500.

Environment
Internal and External

See Also
uc4500

Required License(s)
gateway

```
int uc4511
(
    int chan,
    int len,
    int * sibuff
)
```

int	**chan**	Input	I/O Channel Number returned from uc4500.
int	**len**	Input	Number Of Integers To Read
int *	**sibuff**	Output	Array To Read Integers Into

---

uc4512 (view source)

Defined in: uf_cfi.h

Overview
Read reals from a file opened with uc4500.

Environment
Internal and External

See Also
uc4500

Required License(s)
gateway

```
int uc4512
(
    int chan,
    int len,
    double rbuff [ ]
)
```

int	**chan**	Input	I/O Channel Number returned from uc4500
int	**len**	Input	Number Of Reals To Read
double	**rbuff [ ]**	Output	Array To Read Reals Into

uc4513 (view source)

Defined in: uf_cfi.h

Overview
Read bytes from a file opened with uc4500.

Environment
Internal and External

Required License(s)
gateway

```
int uc4513
(
    int chan,
    int len,
    char bbuf [ UF_MAXWORD ]
)
```

int	**chan**	Input	I/O Channel Number returned by uc4500
int	**len**	Input	Number Of Bytes To Read
char	**bbuf [UF_MAXWORD]**	Output	Array To Read Bytes Into

uc4514a (view source)

Defined in: uf_cfi.h

Overview
Read a line of text from a file opened with uc4504.

Return
Return code
< 0 = Error
>= 0 = Length Of Line Read

Environment
Internal and External

See Also
uc4504

Required License(s)
gateway

```
int uc4514a
(
```

int chan,
char * * cbuf
)

int	**chan**	Input	I/O channel number returned by uc4504
char * *	**cbuf**	Output to UF_*free*	Line read. The buffer must be freed with UF_free()

---

uc4516 (view source)

Defined in: uf_cfi.h

Overview
Randomly read blocks from a file opened with
uc4506. Data is always read on a block boundary.

Return
Return code
< 0 = Error
> 0 = Number of bytes actually read

Environment
Internal and External

See Also
uc4506

Required License(s)
gateway

int uc4516
(
int chan,
int block,
int bytes,
char * cbuf
)

int	**chan**	Input	I/O channel number returned by uc4506
int	**block**	Input	Starting block number to read (from 0)
int	**bytes**	Input	Number of bytes to read
char *	**cbuf**	Output	Data read

---

uc4517 (view source)

Defined in: uf_cfi.h

Overview

Read the next record from a file opened using uc4507.

Environment

Internal and External

See Also

uc4507

Required License(s)

gateway

```
int uc4517
(
    int chan,
    int * bytes,
    char * cbuf
)
```

int	**chan**	Input	I/O channel number returned by uc4507
int *	**bytes**	Output	Length of record read in bytes
char *	**cbuf**	Output	Array to read record into

---

uc4518 (view source)

Defined in: uf_cfi.h

Overview

Read the next directory entry and save the entry's
information in memory. This information may then be accessed by the
routines uc4600 through uc4609.

Return

Return code
< 0 = Error
0 = Entry Read
1 = End Of Subdirectory
2 = End Of Directory

Environment

Internal and External

See Also

uc4508

Required License(s)

gateway

```
int uc4518
(
    void
)
```

uc4519 (view source)

Defined in: uf_cfi.h

Overview
Returns the full filespec of the last directory entry
read. This is provided so the Open C API program need not be
concerned with the syntax of filespecs in forming them.

Environment
Internal and External

See Also
uc4508

Required License(s)
gateway

```
int uc4519
(
    char fspec [ MAX_FSPEC_BUFSIZE ]
)
```

char	**fspec [MAX_FSPEC_BUFSIZE]**	Output	Full file specification of the last directory entry read.

uc4520 (view source)

Defined in: uf_cfi.h

Overview
Write characters to a file opened with uc4500.
The count will be rounded up to the next integer
boundary (e.g. a write of 5 will actually write 8 characters).

Environment
Internal and External

See Also
uc4500

Required License(s)
gateway

```
int uc4520
(
    int chan,
    long len,
    const char * cbuff
)
```

int	**chan**	Input	I/O channel number returned by uc4500.
long	**len**	Input	Number of characters to write
const char *	**cbuff**	Input	Character data to write

---

uc4521 (view source)

Defined in: uf_cfi.h

Overview
Write integers to a file opened with uc4500.

Environment
Internal and External

See Also
uc4500

Required License(s)
gateway

```
int uc4521
(
    int chan,
    long len,
    int * sibuff
)
```

int	**chan**	Input	I/O channel number returned by uc4500
long	**len**	Input	Number of integers to write
int *	**sibuff**	Input	Integer data to write

---

uc4522 (view source)

Defined in: uf_cfi.h

Overview
Write reals to a file opened with uc4500.

Environment
Internal and External

Required License(s)
gateway

```
int uc4522
(
    int chan,
    long len,
    double rbuff [ ]
)
```

int	**chan**	Input	I/O channel number returned by uc4500
long	**len**	Input	Number of reals to write
double	**rbuff [ ]**	Input	Real data to write

---

uc4523 (view source)

Defined in: uf_cfi.h

Overview
Write bytes to a file opened with uc4500.

Environment
Internal and External

See Also
uc4500

Required License(s)
gateway

```
int uc4523
(
    int chan,
    long len,
    const void * bbuff
)
```

int	**chan**	Input	I/O channel number returned by uc4500
long	**len**	Input	Number of bytes to write
const void *	**bbuff**	Input	Byte data to write

---

uc4524 (view source)

Defined in: uf_cfi.h

Overview
Write a line to a text file. Only complete lines may
be written to a text file. The addition of any delimiters (e.g. newline)
is done automatically.

Environment
Internal and External

See Also
uc4504

Required License(s)
gateway

int uc4524
(
int chan,
const char * cbuf
)

int	**chan**	Input	I/O channel number returned by uc4504
const char *	**cbuf**	Input	Line to write

---

uc4525 (view source)

Defined in: uf_cfi.h

Overview
uc4525 is used to write a page break to a text file.

Environment
Internal and External

See Also
uc4504

Required License(s)
gateway

int uc4525
(
int chan
)

int	**chan**	Input	I/O channel number returned by uc4504

---

uc4526 (view source)

Defined in: uf_cfi.h

Overview
Randomly write to a file opened with uc4506. The data is always written on a block boundary.

Environment
Internal and External

See Also
uc4506

Required License(s)
gateway

```
int uc4526
(
    int chan,
    int block,
    int bytes,
    const char * buf
)
```

int	**chan**	Input	I/O channel number returned by uc4506
int	**block**	Input	Starting block number to write (from 0)
int	**bytes**	Input	Number of bytes to write
const char *	**buf**	Input	Data to write

---

uc4527 (view source)

Defined in: uf_cfi.h

Overview
Write a record to a file opened with uc4507. For fixed length record files, the parameter bytes is ignored.

Environment
Internal and External

See Also
uc4507

Required License(s)
gateway

```
int uc4527
(
    int chan,
    int bytes,
    const char * cbuf
)
```

int	**chan**	Input	I/O channel number returned by uc4507
int	**bytes**	Input	Length of record to write in bytes

const char *	**cbuf**	Input	Buffer containing data to write

---

uc4530 (view source)

Defined in: uf_cfi.h

Overview
Skip characters in a file opened with uc4500.
Skips may be backward by using negative skip counts.

Environment
Internal and External

See Also
uc4500

Required License(s)
gateway

```
int uc4530
(
    int chan,
    long len
)
```

int	**chan**	Input	I/O channel number returned by uc4500
long	**len**	Input	Number of characters to skip over

---

uc4531 (view source)

Defined in: uf_cfi.h

Overview
Skip integers in a file opened with uc4500.
Skips may be backward by using negative skip counts.

Environment
Internal and External

Required License(s)
gateway

```
int uc4531
(
    int chan,
    long len
)
```

int	**chan**	Input	I/O channel number returned by uc4500
long	**len**	Input	Number of integers to skip over

---

uc4532 (view source)

Defined in: uf_cfi.h

Overview
Skip reals in a file opened with uc4500.
Skips may be backward by using negative skip counts.

Environment
Internal and External

Required License(s)
gateway

```
int uc4532
(
    int chan,
    long len
)
```

int	**chan**	Input	I/O channel number returned by uc4500
long	**len**	Input	Number of reals to skip over

---

uc4533 (view source)

Defined in: uf_cfi.h

Overview
Skip bytes in a file opened with uc4500.
Skips may be backward by using negative skip counts.

Environment
Internal and External

Required License(s)
gateway

```
int uc4533
(
    int chan,
    long len
)
```

int	**chan**	Input	I/O channel number returned by uc4500
long	**len**	Input	Number of bytes to skip over

---

uc4534 (view source)

Defined in: uf_cfi.h

Overview
Find the current position within a file opened with
uc4500. That position may then be restored using uc4535.

Return
Return code:
< 0 = Error
>= 0 = File position

Environment
Internal and External

See Also
uc4500
uc4535

Required License(s)
gateway

```
long uc4534
(
    int chan
)
```

int	**chan**	Input	I/O channel number returned by uc4500

---

uc4535 (view source)

Defined in: uf_cfi.h

Overview
Restore the read/write position within a file
previously saved using uc4534. The file must have been opened with
uc4500.

Environment
Internal and External

See Also
uc4500
uc4534

Required License(s)

gateway

```
int uc4535
(
    int chan,
    long pos
)
```

int	**chan**	Input	I/O channel number returned by uc4500
long	**pos**	Input	File position returned by uc4534

---

uc4536 (view source)

Defined in: uf_cfi.h

Overview

Reposition a file back to the beginning. This may be used for binary, text, and record I/O files.

Environment

Internal and External

Required License(s)

gateway

```
int uc4536
(
    int chan
)
```

int	**chan**	Input	I/O channel number

---

uc4540 (view source)

Defined in: uf_cfi.h

Overview

Close a file opened with either uc4500, uc4504, uc4506, or uc4507. The close disposition is used to indicate whether a file opened for write is to be saved or not. A normal close will make the file permanent and delete any previous file with the same name. An abort close will delete the file and retain any previous file with the same name.

Environment

Internal and External

Required License(s)

gateway

```
int uc4540
(
    int chan,
    int disp
)
```

int	**chan**	Input	I/O channel number
int	**disp**	Input	Disposition 0 = Normal close 1 = Abort close

---

uc4544 (view source)

Defined in: uf_cfi.h

Overview

Determine characteristics of an open file given its I/O channel number. The integer date/times in qreslt (if qreslt = 6) can be converted to character strings using uc4582.

Starting in NX 11 a query type of 5 is no longer supported.

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4544
(
    int chan,
    int qitem,
    int * qreslt
)
```

int	**chan**	Input	I/O channel number
int	**qitem**	Input	Item to inquire: 1 = File System 2 = File Type 3 = Last Error 4 = Record Format 6 = Creation, Modify, Access Dates
int *	**qreslt**	Output	Query result: For qreslt = 1, 2 = NATIVE For qreslt = 2, See File Types

			For qreslt = 3, Last read or write error For qreslt = 4, = 0 = Variable length records > 0 = Fixed record length For qreslt = 6, qreslt is an array of 6 integers (0)-(1) = Creation Date,Time (2)-(3) = Modification Date,Time (4)-(5) = Last Access Date,Time

---

uc4547 (view source)

Defined in: uf_cfi.h

Overview
Determines the file length (in bytes) of
an open file given its I/O channel number.

Environment
Internal and External

Required License(s)
gateway

```
int uc4547
(
    int chan,
    int qitem,
    int * qreslt
)
```

int	**chan**	Input	I/O channel number
int	**qitem**	Input	Item to query 1 = File length in bytes
int *	**qreslt**	Output	Query result

---

uc4548 (view source)

Defined in: uf_cfi.h

Overview
Close any directories opened with uc4508. If any
subdirectories are open, they will be closed as well.

Environment
Internal and External

See Also
uc4508

Required License(s)

gateway

```
int uc4548
(
    void
)
```

uc4549 (view source)

Defined in: uf_cfi.h

Overview
Close a subdirectory opened with uc4509. Directory
reads will then continue with the previous directory.

Environment
Internal and External

See Also
uc4509

Required License(s)
gateway

```
int uc4549
(
    void
)
```

uc4560 (view source)

Defined in: uf_cfi.h

Overview
Checks whether the specified file of the given type exists.

NOTE: Mixed or upper case file names may not be found if the
environment variable UGII_OPTION = LOWER is set.

Passing an ftype of 0 will look for a file but does not work for a directory.
To check for a directory the ftype must be set to 100.

Return
Return code:
< 0 = Error
= 0 = File Exists
= 1 = File Does Not Exist

Environment
Internal and External

See Also
For description of file types see table

Required License(s)
gateway

int uc4560
(
  const char * fspec,
  int ftype
)

const char *	**fspec**	Input	File to check
int	**ftype**	Input	File type 0 will check for files 100 will check for directories

---

uc4561 (view source)

Defined in: uf_cfi.h

Overview
Remove a given file from the file system.

Environment
Internal and External

See Also
For description of file types see table

Required License(s)
gateway

int uc4561
(
  const char * fspec,
  int ftype
)

const char *	**fspec**	Input	File to delete
int	**ftype**	Input	File type

---

uc4562 (view source)

Defined in: uf_cfi.h

Overview

Change the name of a given file. The new file name should be a simple name (e.g. no directory specification).

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4562
(
    const char * fspec,
    int ftype,
    const char * fspec2
)
```

const char *	**fspec**	Input	Old file name
int	**ftype**	Input	File type
const char *	**fspec2**	Input	New file name

---

uc4563 (view source)

Defined in: uf_cfi.h

Overview

Create an empty directory.

Environment

Internal and External

Required License(s)

gateway

```
int uc4563
(
    const char * fspec,
    int ftype
)
```

const char *	**fspec**	Input	Directory to create
int	**ftype**	Input	File type

---

uc4564 (view source)

Defined in: uf_cfi.h

Overview

Retrieve the header information of a single file and
store it in memory. The information may then be retrieved using
uc4600 through uc4609.

Environment

Internal and External

See Also

uc4600
uc4601
uc4602
uc4603
uc4605
uc4606
uc4607
uc4608
uc4609
For description of file types see table

Required License(s)

gateway

```
int uc4564
(
    const char * fspec,
    int ftype,
    int fmode
)
```

const char *	**fspec**	Input	File specification from which to read header
int	**ftype**	Input	File type
int	**fmode**	Input	Bit-mask Specifying Header Fileds Desired bit 0 = Read Owner Field 0 = No 1 = Yes bit 1 = Read Protection Class Field 0 = No 1 = Yes bit 2 = Read Status Field 0 = No 1 = Yes bit 3 = Read Length Field 0 = No 1 = Yes bit 4 = Read Dates 0 = No 1 = Yes bit 5 = Read Description Field 0 = No 1 = Yes bit 6 = Read Customer Area Field 0 = No 1 = Yes bit 7 = Read Machine Field 0 = No

			1 =yes bit 8-15 = Reserved

uc4565 (view source)

Defined in: uf_cfi.h

Overview
Read the current default value for a directory.

Environment
Internal and External

Required License(s)
gateway

```
int uc4565
(
    int def,
    char fspec [ MAX_FSPEC_BUFSIZE ]
)
```

int	**def**	Input	Default to read: 1 = Current directory 2 = '$' directory 3 = '!' directory
char	**fspec [MAX_FSPEC_BUFSIZE]**	Output	Current Setting

uc4566 (view source)

Defined in: uf_cfi.h

Overview
Change the current user's directory.

Environment
Internal and External

Required License(s)
gateway

```
int uc4566
(
    int def,
    const char * fspec
)
```

int	**def**	Input	Default to change: 1 = Current directory

			2 = '$' directory 3 = '!' directory
const char *	**fspec**	Input	New default value

uc4567 (view source)

Defined in: uf_cfi.h

Overview

Copies or moves a file from a source file specification to a destination file specification. When any move option is used, the source file will only be deleted after it has been successfully copied to the destination file.

Using a file type of -1 indicates "any file type" so changing extensions during the copy or move operation will work correctly.

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4567
(
    const char * srcspc,
    const char * dstspc,
    int cmode,
    int stype,
    int dtype
)
```

const char *	**srcspc**	Input	Source file specification
const char *	**dstspc**	Input	Destination file specification
int	**cmode**	Input	Specifies Copy/Move Options: = UF_CFI_COPY_NEVER_REPLACE = UF_CFI_COPY_ALWAYS_REPLACE = UF_CFI_COPY_REPLACE_IF_NEWER = UF_CFI_COPY_LEGACY (same as UF_CFI_COPY_NEVER_REPLACE) = UF_CFI_MOVE_NEVER_REPLACE = UF_CFI_MOVE_ALWAYS_REPLACE = UF_CFI_MOVE_REPLACE_IF_NEWER = UF_CFI_MOVE_LEGACY (same as UF_CFI_MOVE_NEVER_REPLACE)
int	**stype**	Input	Source file type
int	**dtype**	Input	Destination file type

uc4570 (view source)

Defined in: uf_cfi.h

Overview

Examine a filespec and make sure it conforms to the syntax of the file system.

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4570
(
    const char * fspec,
    int ftype
)
```

const char *	**fspec**	Input	File specification to validate
int	**ftype**	Input	File type

---

uc4571 (view source)

Defined in: uf_cfi.h

Overview

Examine a filespec and make sure it is a valid directory specification for the file system. A "directory file spec" is a path to a file which is a directory.

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4571
(
    const char * fspec,
    int ftype
)
```

const char *	**fspec**	Input	Directory File Specification To Validate

int	**ftype**	Input	File Type

---

uc4572 (view source)

Defined in: uf_cfi.h

Overview
Examine a filespec and make sure it is a valid wildcard directory specification
for the file system. This will also indicate whether there were any
wildcard characters in the filespec. To find all files that match the wildcard
filespec, open the directory with uc4508 and supply the wildcard filespec,
then read (uc4518) each matching entry and the corresponding filespec (uc4519).

Return
Return code:
= 0 = Valid file specification
= 1 = Valid with wildcards
Anything else is an error

Environment
Internal and External

See Also
For description of file types see table

Required License(s)
gateway

```
int uc4572
(
    const char * fspec,
    int ftype
)
```

const char *	**fspec**	Input	File specification to validate
int	**ftype**	Input	File type

---

uc4573 (view source)

Defined in: uf_cfi.h

Overview
Given a filespec return its fully qualified equivalent. If
a non-zero file type is given, the extension will also be set. For
example, the native system with a default directory DISK2/JOE
fspec = "foo", type = 2 will produce expfspec = "DISK2/JOE/FOO.PRT"

Environment
Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4573
(
    const char * fspec,
    int ftype,
    char expfspec [ MAX_FSPEC_BUFSIZE ]
)
```

const char *	**fspec**	Input	File specification to expand
int	**ftype**	Input	File type
char	**expfspec [MAX_FSPEC_BUFSIZE]**	Output	Expanded file specification

---

uc4574 (view source)

Defined in: uf_cfi.h

Overview

Accept a filespec and removes any directory path, extension,
and any system specific information and returns the resultant simple
filename. For example:
fspec = "/DISK1/JOE/FOO.PRT", ftype = 2 will produce fname = "FOO".

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4574
(
    const char * fspec,
    int ftype,
    char fname [ UF_CFI_MAX_FILE_NAME_BUFSIZE ]
)
```

const char *	**fspec**	Input	File specification from which to extract name
int	**ftype**	Input	File type
char	**fname [UF_CFI_MAX_FILE_NAME_BUFSIZE]**	Output	Simple file name

uc4575 (view source)

Defined in: uf_cfi.h

Overview

Combine a directory with a filename producing a file specification (filespec).
For example:

dspec = "/manager", ftype = 2, fname = "bar"

will produce fspec = "/MANAGER/BAR.PRT".

If the file name is a directory, using a filetype of 100 will merge the directories. For example:

dspec = "/manager", ftype = 100, fname = "bar"

will produce fspec = "/MANAGER/BAR".

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4575
(
    const char * dspec,
    int ftype,
    const char * fname,
    char fspec [ MAX_FSPEC_BUFSIZE ]
)
```

const char *	**dspec**	Input	Directory
int	**ftype**	Input	File type
const char *	**fname**	Input	File name
char	**fspec [MAX_FSPEC_BUFSIZE]**	Output	Resultant file specification

uc4576 (view source)

Defined in: uf_cfi.h

Overview

Take a filespec and returns its directory and filename components. For example:

fspec = "/Manager/Work/BENCH", ftype = 2

Note that this routine is impacted by the UGII_OPTION environment variable,
so if UGII_OPTION=lower is set, then the above example will produce:
dspec = "/manager/work" and fname = "bench.prt".

Environment
Internal and External

See Also
For description of file types see table

Required License(s)
gateway

```
int uc4576
(
    const char * fspec,
    int ftype,
    char dspec [ MAX_FSPEC_BUFSIZE ] ,
    char fname [ UF_CFI_MAX_FILE_NAME_BUFSIZE ]
)
```

const char *	**fspec**	Input	File specification to split up
int	**ftype**	Input	File type
char	**dspec [MAX_FSPEC_BUFSIZE]**	Output	Directory component
char	**fname [UF_CFI_MAX_FILE_NAME_BUFSIZE]**	Output	File name component

---

uc4577 (view source)

Defined in: uf_cfi.h

Overview
The name returned is a unique name for a temporary file. The resultant
filename is unique from other processes at the time. From a single
process, filenames will begin duplicating after the first 1,679,615 calls
to uc4577. Temporary files should be deleted when no longer needed by an
application. If the files are not deleted, there is a chance that the same
name may come up again if the same user happens to get the same process id
on a later date.

The maximum number of characters which will be returned is 12.

Environment
Internal and External

Required License(s)
gateway

```
int uc4577
(
    char fname [ UF_MAX_UNIQUE_FILE_NAME_BUFSIZE ]
```

)

char	**fname [UF_MAX_UNIQUE_FILE_NAME_BUFSIZE]**	Output	Unique filename

uc4578 (view source)

Defined in: uf_cfi.h

Overview
Remove the file extension from a given file specification and
returns the resultant file specification.

Environment
Internal and External

See Also
For description of file types see table

Required License(s)
gateway

```
int uc4578
(
    const char * fspec,
    int ftype,
    char dspec [ MAX_FSPEC_BUFSIZE ]
)
```

const char *	**fspec**	Input	File specification
int	**ftype**	Input	File type
char	**dspec [MAX_FSPEC_BUFSIZE]**	Output	Resultant file specification

uc4579 (view source)

Defined in: uf_cfi.h

Overview
Form the full filespec, given a simple name of a file
in the UGII_UTIL directory.

Environment
Internal and External

See Also
For description of file types see table

Required License(s)
gateway

```
int uc4579
(
    const char * fname,
    int ftype,
    char fspec [ MAX_FSPEC_BUFSIZE ]
)
```

const char *	**fname**	Input	File name
int	**ftype**	Input	File type
char	**fspec [MAX_FSPEC_BUFSIZE]**	Output	Resultant file specification

---

uc4580 (view source)

Defined in: uf_cfi.h

Overview

return the four character symbolic name for a given ftype
code (e.g.: 'PART' for ftype code 2). Many ftype codes will return
'TEXT' which indicates the file's contents may be displayed as ascii
data. Unnamed ftypes will have their numeric code returned in ascii.

Return

Return code:
< 0 = Error
= 0 = Format Returned
= 1 = Format Undefined

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4580
(
    int ftype,
    char symb [ 5 ]
)
```

int	**ftype**	Input	File type
char	**symb [5]**	Output	Symbolic name

---

uc4581 (view source)

Defined in: uf_cfi.h

Overview

Convert the symbolic character representation of a file type
into its numeric equivalent. For example, "PART" translates to a 2.

Return

Return code:
= 0 = Unknown Symbolic Name
> 0 = File Type

Environment

Internal and External

Required License(s)

gateway

```
int uc4581
(
    const char * symb
)
```

const char *	**symb**	Input	Symbolic file type

---

uc4582 (view source)

Defined in: uf_cfi.h

Overview

Convert NX computational time to display form. Two forms are available
for the date and two forms are available for the time.
A date or time of -1 returns the current date and/or time.

NOTE: In option 9 of the dtype argument, "formatted for the locale"
means that the date and time string is appropriate for the language in
which the user's operating system environment runs under.

Return

Return code:
0 = No error
not 0 = Error code

Environment

Internal and External

See Also

uc4583

History

The dtype argument was modified in V13.0 to
increase the number of options from 4 to 9.

Required License(s)

gateway

```
int uc4582
(
    int date [ 2 ] ,
    int dtype,
    char date_string [ 21 ] ,
    char time [ 21 ]
)
```

int	**date [2]**	Input	Computational time: [0] Date [1] Time
int	**dtype**	Input	Date And time representation: 1 = mm/dd/yy, hh:mm 2 = mm/dd/yy, hh:mm xM 3 = dd-mmm-yy, hh:mm 4 = dd-mmm-yy, hh:mm xM 5 = mm/dd/yyyy, hh:mm 6 = mm/dd/yyyy, hh:mm xM 7 = dd-mmm-yyyy, hh:mm 8 = dd-mmm-yyyy, hh:mm xM 9 = Formatted for the locale where 'mm' = numeric month, 'dd' = day, 'yy' = two digit year, 'yyyy' = four digit year, 'mmm' = symbolic month, 'hh' = hour, 'mm' = minute, 'x' = 'A' or 'P' When a blank is passed in for 'x', dtype = 8 will default to 12 hour time format where dtype = 7 will display a 24 hour time format. Note: On Windows any string can be specified as AM/PM by using Control Panel -> Regional and Language Options -> Customize -> Regional Options -> Time
char	**date_string [21]**	Output	Date (20 characters max)
char	**time [21]**	Output	Time (20 characters max)

---

uc4583 (view source)

Defined in: uf_cfi.h

Overview

Convert a character date and time to NX computational date and time.

Return

Return Code:
0 = Success
1 = Failure

Environment

Internal and External

See Also

uc4582

Required License(s)

gateway

int uc4583
(
const char * date,
const char * time,
int* dandt
)

const char *	**date**	Input	Date in any of the following forms MM/DD/YY DD-MMM-YY DDMMMYY MM/DD/YYYY DD-MMM-YYYY DDMMMYYYY If date is blank, the current date is used
const char *	**time**	Input	Time in either of the following forms HH:MM HH:MM xM (x = 'A' or 'P') If time is blank, the current time is used
int*	**dandt**	Output	Date And Time (1) = Computational Date (2) = Computational Time

---

uc4595 (view source)

Defined in: uf_cfi.h

Overview

Query the user name, String Result

Environment

Internal and External

Required License(s)

gateway

int uc4595
(
int qitem,
char str [17]
)

int	**qitem**	Input	Item to query: 1 = Username
char	**str [17]**	Output	Query result. This must be a buffer big enough to hold the user name.

uc4596 (view source)

Defined in: uf_cfi.h

Overview

Query a set of characteristics. The result for each item code follows:
Login Status: (qitem = 1)
bit 0 = Login Status
0 = NOT LOGGED IN
1 = LOGGED IN
bit 1 = Username Status
0 = DO NOT NEED A USERNAME TO LOGIN
1 = USERNAME NEEDED FOR LOGIN
bit 2 = Password Status
0 = DO NOT NEED A PASSWORD TO LOGIN
1 = PASSWORD NEEDED FOR LOGIN
bits 3-15 = Reserved
File Header Support: (qitem = 2)
bit 0 = Owner Supported
0 = NO
1 = YES
bit 1 = Protection Classes Supported
0 = NO
1 = YES
bit 2 = Status Word Supported
0 = NO
1 = YES
bit 3-4 = Reserved
bit 5 = Description Supported
0 = NO
1 = YES
bit 6 = Customer Area Supported
0 = NO
1 = YES
bit 7 = Non-Native Files Supported
0 = NO
1 = YES
bits 8-15 = Reserved

Environment

Internal and External

Required License(s)

gateway

```
int uc4596
(
    int qitem,
    int * qreslt
)
```

int	**qitem**	Input	Item to query 1 = Login status 2 = File header fields supported
int *	**qreslt**	Output	Query result

---

uc4599 (view source)

Defined in: uf_cfi.h

Overview
Translate an error code to the text associated with it.
Due to the way error handling is done in the file system routines, the
error text should be retrieved before another error occurs otherwise
the error message might be lost.

Environment
Internal and External

Required License(s)
gateway

```
int uc4599
(
    int ug_errorno,
    char errstg [ MAX_LINE_BUFSIZE ]
)
```

int	**ug_errorno**	Input	Error Code
char	**errstg [MAX_LINE_BUFSIZE]**	Output	Error Text

---

uc4600 (view source)

Defined in: uf_cfi.h

Overview
Return the simple file name of the last file read with uc4518 or uc4564..
To obtain the full file specification, including the directory use uc4519.

Environment
Internal and External

See Also
uc4518
uc4564
uc4519

Required License(s)
gateway

```
int uc4600
(
    char fname [ UF_CFI_MAX_FILE_NAME_BUFSIZE ]
)
```

char	**fname [UF_CFI_MAX_FILE_NAME_BUFSIZE]**	Output	Filename

---

uc4601 (view source)

Defined in: uf_cfi.h

Overview
Return the file type of the last file read with uc4518 or uc4564..
You may use uc4581 to translate the file type to a character string.

Return
Return code:
< 0 = error
>= 0 = file type

Environment
Internal and External

See Also
uc4518
uc4564
uc4581

Required License(s)
gateway

```
int uc4601
(
    void
)
```

---

uc4602 (view source)

Defined in: uf_cfi.h

Overview
Return the status word of the last file read with uc4518 or uc4564..

Environment
Internal and External

See Also
uc4518
uc4564

Required License(s)
gateway

```
int uc4602
(
    int * fsts
)
```

int *	**fsts**	Output	Status word

---

uc4603 (view source)

Defined in: uf_cfi.h

Overview
Return the owner of the last file read with uc4518 or uc4564..

Environment
Internal and External

See Also
uc4518
uc4564

Required License(s)
gateway

```
int uc4603
(
    char owner [ 17 ]
)
```

char	**owner [17]**	Output	Owner of file (16 characters max)

---

uc4605 (view source)

Defined in: uf_cfi.h

Overview
Return the length of the last file read with uc4518 or uc4564..

Return
Return code:
>= 0 = File Length In Bytes
< 0 = error code

Environment
Internal and External

See Also
uc4518
uc4564

Required License(s)
gateway

```
long uc4605
(
    void
)
```

uc4606 (view source)

Defined in: uf_cfi.h

Overview
Return the creation, modification, and last access date/time
of the last file read with uc4518 or uc4564..
Use uc4582 to convert the date/time to character strings.

Environment
Internal and External

See Also
uc4518
uc4564
uc4582

Required License(s)
gateway

```
int uc4606
(
    int * cdate,
    int * mdate,
    int * ldate
)
```

int *	**cdate**	Output	Two word array containing the creation date and time
int *	**mdate**	Output	Two word array containing the modification date and time
int *	**ldate**	Output	Two word array containing the last access date and time

uc4607 (view source)

Defined in: uf_cfi.h

Overview
Return the descriptions area of the last file read with uc4518 or uc4564..

Environment
Internal and External

See Also

uc4518
uc4564

Required License(s)

gateway

```
int uc4607
(
    char darea [ MAX_LINE_BUFSIZE ]
)
```

char	**darea [MAX_LINE_BUFSIZE]**	Output	Description Area (132 characters max)

---

uc4608 (view source)

Defined in: uf_cfi.h

Overview

Return the customer area of the last file read with uc4518 or uc4564..

Environment

Internal and External

See Also

uc4518
uc4564

Required License(s)

gateway

```
int uc4608
(
    char carea [ MAX_LINE_BUFSIZE ]
)
```

char	**carea [MAX_LINE_BUFSIZE]**	Output	Customer area (132 characters max)

---

uc4609 (view source)

Defined in: uf_cfi.h

Overview

Return the machine field of the last file read with uc4518 or uc4564.
These values are available for part files only. Parts filed in
V10.0 or earlier return unknown values.

Return

Return code:
< 0 = error
1 = APOLLO

2 = DEC VAX/VMS
3 = HP CISC
4 = HP RISC
5 = SUN 3
6 = SUN SPARC
7 = DEC RISC (ULTRIX)
8 = SGI
9 = DATA GENERAL
10 = IBM MVS
11 = IBM AIX
12 = AXP/OSF
13 = AXP/VMS

Environment
Internal and External

See Also
uc4518
uc4564

Required License(s)
gateway

```
int uc4609
(
    void
)
```

---

uc4612 (view source)

Defined in: uf_cfi.h

Overview
Modify the status header field of a file. An error returns if the file system does not support a status field or the user does not have the privilege to change it.

Environment
Internal and External

See Also
For description of file types see table

Required License(s)
gateway

```
int uc4612
(
    const char * fname,
    int ftype,
    int fsts
)
```

const char *	**fname**	Input	File name

int	**ftype**	Input	File type
int	**fsts**	Input	New status value

---

uc4613 (view source)

Defined in: uf_cfi.h

Overview

Modify the owner header field of a file. An error returns if the file system does not support an owner field or the user does not have the privilege to change it (some operating systems may require root privilege to change file ownership).

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4613
(
    const char * fname,
    int ftype,
    const char * owner
)
```

const char *	**fname**	Input	File name to change the owner of
int	**ftype**	Input	File type
const char *	**owner**	Input	New owner value

---

uc4617 (view source)

Defined in: uf_cfi.h

Overview

Change the description header field of a file. An
error is returned if the file system does not support a description field or
the user does not have the privilege to change it.

Environment

Internal and External

See Also

For description of file types see table

Required License(s)

gateway

```
int uc4617
(
    const char * fname,
    int ftype,
    const char * desc
)
```

const char *	**fname**	Input	File name to change description of
int	**ftype**	Input	File type
const char *	**desc**	Input	New description value

---

uc4618 (view source)

Defined in: uf_cfi.h

Overview

Change the customer area header field of a file. An
error will be returned if the file system does not support a customer
area field or the user does not have the privilege to change it.
This function modifies the part file on disk and should not
be used on a part file that has already been opened in NX. If this
occurs, the part cannot be saved. Use UF_PART_set_customer_area to
modify the customer area of a loaded part.

Environment

Internal and External

See Also

UF_PART_set_customer_area
For description of file types see table

Required License(s)

gateway

```
int uc4618
(
    const char * fname,
    int ftype,
    const char * carea
)
```

const char *	**fname**	Input	File name to change the customers area of
int	**ftype**	Input	File type
const char *	**carea**	Input	New customer area value

---

uc4620 (view source)

Defined in: uf_cfi.h

Overview

Read a switch from the program command line given the name of the switch.
All switches are global switches; they may appear anywhere on the command line.
Switches may have a value or no value. For example:
/LIST=FOO.LIS switch with a value
/LIST switch with no value

Switches must match completely. If sname is "USERNAME", you
must enter the full text string.

Under UNIX, switches take the form:
-name no value
-name=value switch with a value

Switches are separated by blanks on UNIX. For example:
ugraf -user=manager -pass=frogs

Under WNT, switches take the form:
-name no value
-name:value switch with a value
-name=value switch with a value

NOTE: Use uc4624 in conjunction with this function. You must call
uc4624 before calling either uc4620 or uc4621.

Return

Return code:
< 0 = Error
0 = Switch Not Present
1 = Switch Found With No Value
3 = Switch Found With A Value

Environment

External

See Also

uc4624
uc4621

Required License(s)

gateway

```
int uc4620
(
    const char * sname,
    char swstg [ MAX_LINE_BUFSIZE ]
)
```

const char *	**sname**	Input	Switch name
char	**swstg [MAX_LINE_BUFSIZE]**	Output	Switch value

uc4621 (view source)

Defined in: uf_cfi.h

Overview
Read arguments from the command line. Each argument may be read only once.
NOTE: Use uc4624 in conjunction with this function. You must call
uc4624 before calling either uc4620 or uc4621.

Return
Return code:
< 0 = Error
0 = Argument Not Present
1 = Argument Found

Environment
External

See Also
uc4624
uc4620

Required License(s)
gateway

```
int uc4621
(
    char nxtarg [ MAX_FSPEC_BUFSIZE ]
)
```

char	**nxtarg [MAX_FSPEC_BUFSIZE]**	Output	Argument Value

---

uc4622 (view source)

Defined in: uf_cfi.h

Overview
Returns an argument list to the GRIP xspawn command. The returned string
can not exceed 132 characters.

Environment
Internal and External

Required License(s)
gateway

```
int uc4622
(
    char * ip1
)
```

char *	**ip1**	Input	The return argument list

uc4623 (view source)

Defined in: uf_cfi.h

Overview

Returns a pointer to a string which is the release number of the specified part file. Some of the possible values are shown for the description of cr2. You must allocate sufficient size for the relnum array. For example, you could use MAX_FSPEC_BUFSIZE (prototyped in uf_defs.h) for the size of the array.

Environment

Internal and External

Required License(s)

gateway

```
int uc4623
(
    const char * fspec,
    char relnum [ 133 ]
)
```

const char *	**fspec**	Input	Part file name
char	**relnum [133]**	Output	Release number An example of possible Return Values are: V8, V9, V10, V10.1, V10.2, V10.3 etc.

uc4624 (view source)

Defined in: uf_cfi.h

Overview

Save argument names for use with uc4620 and uc4621. The prog parameter is not used. It is only present for backward compatibility. You must use uc4624 before calling uc4620 or uc4621.

Note that uc4624 expects to receive the argc and argv values passed to the program from main, and so the values in argv are assumed to be in the current users locale not UTF8 data. As such this routine does not honor the setting of the text mode made by calling UF_TEXT_set_text_mode().

Environment

External

See Also

uc4620
uc4621

Required License(s)

gateway

```
int uc4624
(
    int prog,
    int argc,
    char * * argv
)
```

int	**prog**	Input	Not used
int	**argc**	Input	Argument count
char * *	**argv**	Input	array of argument names. This data is always assumed to be in the users current locale.

---

uc4650 (view source)

Defined in: uf_cfi.h

Overview

Outputs a sorted directory listing to the Information Window if it has been opened. Use UF_UI_open_listing_window to open the Information Window. Dates in cbuf must be in the format DD-MMM-YY (eg. 04-JUL-89).

Return

Return code:
< 0 = Error code
0 = No files listed
> 0 = Number of files listed

Environment

UF_UI_open_listing_window

Required License(s)

gateway

```
int uc4650
(
    const char * dir,
    int fmode,
    int smode,
    int * pbuf,
    int * ibuf,
    const char * cbuf
)
```

const char *	**dir**	Input	Input directory
int	**fmode**	Input	File selection mode 1 = Select all files 2 = File template specified in cbuf 3 = Select file created/modified/ accessed after date specified in cbuf according to field specified in ibuf

			4 = Select file created/modified/ accessed before date specified in cbuf according to field specified in ibuf 5 = Select file by owner specified in cbuf 6 = Select file by protection class specified in cbuf 7 = Select files of type specified in ibuf 8 = Select files by status specified in ibuf
int	**smode**	Input	Sort mode 1 = Alphabetic 2 = Creation date 3 = Modified date 4 = Access date 5 = Owner
int *	**pbuf**	Input	Print Field Selection Array, Set Array Element = 1 To Print Desired Field (1) = Print format (2) = Print owner (3) = Print pclass (4) = Print length (5) = Print status (6) = Print creation date (7) = Print creation time (8) = Print modification date (9) = Print modification time (10) = Print access date (11) = Print access time (12) = Print machine type (13) = Print description area (14-16) = Reserved
int *	**ibuf**	Input	Integer Parameter Array IF fmode=3 or fmode=4, ibuf show date type IF ibuf(1) = 1 : use file creation date IF ibuf(1) = 2 : use file modified date IF ibuf(1) = 3 : use file accessed date IF fmode=7, ibuf selects file type set array element = 1 to select desired file type (1) = Part (2) = Symbol (3) = Text (4) = GRIP (5) = Customer (6) = UNISOLIDS (7) = UGI (8) = Communications (9) = Keystroke (10) = Display (11) = CL file (12) = Directory (13-16) = Reserved IF fmode=8, ibuf(1) = File Status
const char *	**cbuf**	Input	Character Parameter IF fmode=2, cbuf contains file template IF fmode=3, cbuf contains file date IF fmode=4, cbuf contains file date IF fmode=5, cbuf contains owner name IF fmode=6, cbuf contains protection class

uc4901 (view source)

Defined in: uf_cfi.h

Overview

Return the language name stored in the native binary file.
You can use the returned language name string
to differentiate languages that use the same character set.
NOTE: If a Native Binary File has not been loaded then uc4901
returns "ENGLISH".

Return

Return code
0 = No error
not 0 = Error code

Environment

Internal and External

Required License(s)

gateway

```
int uc4901
(
    char lname [ MAX_FSPEC_BUFSIZE ]
)
```

char	**lname [MAX_FSPEC_BUFSIZE]**	Output	Returns The Language Name Stored In The Native Binary File

UF_CFI_ask_file_exist (view source)

Defined in: uf_cfi.h

Overview

Test if a file exists.

Note: This function only works with files - not directories. To check if a directory exists use uc4560 and pass a file type of 100.

Return

0 - No error
Otherwise - Error Code

Environment

Internal and External

History

Originally released in V16.0

Required License(s)

gateway

```
int UF_CFI_ask_file_exist
(
    const char * file_spec,
    int * status
)
```

const char *	**file_spec**	Input	The file to check
int *	**status**	Output	File existence status. 0 - file exists 1 - file does not exist

---

UF_CFI_spawn (view source)

Defined in: uf_cfi.h

Overview

Spawn a subprocess. The return code will indicated the status of the process creation. If the status from the actual command is needed, use UF_CFI_spawn_check_status.

Environment

Internal and External

See Also

UF_CFI_spawn_check_status

History

Originally released in V18.0

Required License(s)

gateway

```
int UF_CFI_spawn
(
    const char * program,
    int num_args,
    char * arguments [ ] ,
    logical is_concur,
    int * process_id
)
```

const char *	**program**	Input	The command to be executed. This command must either be a full path name, or the program must be found on the path.
int	**num_args**	Input	The number of arguments in the next array. These arguments will be passed to the command.
char *	**arguments [ ]**	Input	An array of character pointers for the arguments to be passed to the program. You may pass in a NULL if there are not any arguments. These arguments will be added in the order they are stored, so the command will be: program argument[0] argument[1] ... Switches must be formatted by the caller. On NT, switches take the form "-switch:value", where on Unix switches take the form "-switch=value".

logical	**is_concur**	Input	If TRUE, the command will be run at the same time as the NX Open program, if FALSE, then UF_CFI_spawn will wait for the completion of the command prior to returning to the caller.
int *	**process_id**	Output	The process ID of the spawned process. This process ID can be used to check the status of the spawned process using UF_CFI_spawn_check_status.

UF_CFI_spawn_check_status (view source)

Defined in: uf_cfi.h

Overview
Check the status of a spawned subprocess.

Environment
Internal and External

See Also
UF_CFI_spawn

History
Originally released in V18.0

Required License(s)
gateway

```
int UF_CFI_spawn_check_status
(
    int process_id,
    logical * still_running,
    int * return_status
)
```

int	**process_id**	Input	The process id returned by UF_CFI_spawn for the command that was run. Note that this is only returned for processes that are run concurrently.
logical *	**still_running**	Output	If TRUE, the command is still running. If FALSE, the command has completed.
int *	**return_status**	Output	If still_running is FALSE, then this is the return status from the child process. If still_running is TRUE, then this will be set to zero. A return_status of 127 is set when the spawned command could not be found.
