Acorn Disc Filing System - ADFS

Filesystems supported by ADFS

The ADFS module supports the following Filecore formats which have:

- new maps
- new directories or big directories

In terms of the named formats, this means we support:

- E and E+, with or without boot block
- F and F+

We fully support reading files from these filesystems, and writing to existing files within their existing allocation. Essentially, we do not support changing any of the filesystem metadata.

This is intended to support loopback mounted Linux native filesystems on a RISC OS Filecore filesystem, but will allow the data within files to be changed.

If write support (ADFS_FS_RW) is configured, we allow rudimentary directory updates, specifically updating the access mode and timestamp.

Mount options for ADFS

uid=nnn	All files in the partition will be owned by user id nnn. Default 0 (root).
gid=nnn	All files in the partition will be in group nnn. Default 0 (root).
ownmask=nnn	The permission mask for ADFS 'owner' permissions will be nnn. Default 0700.
othmask=nnn	The permission mask for ADFS 'other' permissions will be nnn. Default 0077.
ftsuffix=n	When ftsuffix=0, no file type suffix will be applied. When ftsuffix=1, a hexadecimal suffix corresponding to the RISC OS file type will be added. Default 0.

Mapping of ADFS permissions to Linux permissions

ADFS permissions consist of the following:

- Owner read
- Owner write
- Other read
- Other write

(In older versions, an 'execute' permission did exist, but this does not hold the same meaning as the Linux 'execute' permission and is now obsolete).

The mapping is performed as follows:

```
    Owner read                          -> -r--r--r--
    Owner write                         -> --w--w---w
    Owner read and filetype UnixExec    -> ---x--x--x
These are then masked by ownmask, eg 700 -> -rwx------
    Possible owner mode permissions     -> -rwx------

    Other read                          -> -r--r--r--
    Other write                         -> --w--w--w-
    Other read and filetype UnixExec    -> ---x--x--x
These are then masked by othmask, eg 077 -> ----rwxrwx
    Possible other mode permissions     -> ----rwxrwx
```

Hence, with the default masks, if a file is owner read/write, and not a UnixExec filetype, then the permissions will be:

```
-rw-------
```

However, if the masks were ownmask=0770,othmask=0007, then this would be modified to:

```
-rw-rw----
```

There is no restriction on what you can do with these masks. You may wish that either read bits give read access to the file for all, but keep the default write protection (ownmask=0755,othmask=0577):

```
-rw-r--r--
```

You can therefore tailor the permission translation to whatever you desire the permissions should be under Linux.

RISC OS file type suffix

RISC OS file types are stored in bits 19..8 of the file load address.

To enable non-RISC OS systems to be used to store files without losing file type information, a file naming convention was devised (initially for use with NFS) such that a hexadecimal suffix of the form ,xyz denoted the file type: e.g. BasicFile,ffb is a BASIC (0xffb) file. This naming convention is now also used by RISC OS emulators such as RPCEmu.

Mounting an ADFS disc with option ftsuffix=1 will cause appropriate file type suffixes to be appended to file names read from a directory. If the ftsuffix option is zero or omitted, no file type suffixes will be added.