

I.2-UNIX-OFS PROGRAM EXECUTION INFORMATION FOR OPERATIONAL FORECAST SYSTEM PROGRAMS ON UNIX SYSTEMS

Scripts

The script ofs can be used to execute the following Operational Forecast System programs:

- ESPINIT
- FCINIT
- FCST
- FILECRAT
- FILESIZE
- GOESDB
- PPDUTIL
- PPINIT
- PRDUTIL
- REORDER
- SASMDB
- SHEFPARS
- SHEFPOST

The command format is:

```
ofs -p progname
    [-d]
    [-i in_file] [-o out_file_prefix]
    [-f fileset] [-u user] [-g input_group] [-r reor_set]
    [-t] [-x]
```

The only required parameter is the program to be executed indicated by the -p option.

All other parameters are optional and will be provided values if not supplied on the command line. Available options are:

<u>Option</u>	<u>Description</u>	<u>Default Value</u>
-d	Use development executable directory indicated by the token my_rls	Use directory specified by the token ofs_rls
-f	OFS file set override	The file set specified by the token ofs_level
-g	Input file group override	The group specified by the token ofs_inpt_grp
-i	Input file to be used with program - file must be located in the appropriate programs input directory	None
-o	Output file prefix or 'tty' - output files are date-time	progname

<u>Option</u>	<u>Description</u>	<u>Default Value</u>
	stamped and are placed in the user's output directory - if 'tty' is specified output is written to the terminal	
-p	Program name	None
-r	Reorder file set override	The file set specified by the token ofs_reor_lvl
-t	Output log information displayed to the terminal	Output to log file
-u	User name override - used to place output in output directory other than the submitting user's	Login user name (\$LOGNAME)
-x	Conduct execution check only displaying additional information - program is not executed	Program will be executed

For example, program FCINIT can be run with the script ofs as follows:

```
ofs -p espinit -i defseg -o defseg
```

or

```
ofs -p espinit -i defseg -o defseg -d
```

where '-d' specifies executable directory \$my_rls

FCINIT input files must be in the user's forecast directory in a subdirectory named espinit. For example:

```
.../nwsrfs/ofs/input/oper/fcinit
```

The script fs5fcopy is used to copy from one set of OFS files to another. The command format is:

```
fs5fcopy -f user_in -t user_out
```

where user_in is the name of the input files set
user_out is the name of the output files set

Apps Defaults Tokens

The Operational Forecast System scripts and programs the Apps Default System to set execution controls and path names (see Chapter I.2-UNIX-APPSDFLT).

FCST Program MODS Management

MODS are stored in a file in the directory

```
<ofs_dir>/files/<ofs_level>/mods
```

where <ofs_dir> is the OFS file directory
<ofs_level> is the set of OFS files being used

The name specified on the .INCLUDE statement is the file name in the directory.

The Interactive Forecast Program works on a Forecast Group basis and will read any existing MODs in the file named after the Forecast Group. It also places the MODs generated from the session back into the same file. Therefore, the file name should be the same as the Forecast Group name.

Reordering OFS Files

The script ofs_reorder is used to reorder the OFS files. The command format is:

```
ofs_reorder [user_in] [user_out]
```

where user_in is the name of the input file set
user_out is the name of the output file set

ofs_reorder runs the following programs:

- o FILECRAT to create the new OFS files
- o PRDUTIL to define the data types in the new OFS files
- o REORDER to reorder the old OFS files to the new OFS files

The following steps should be performed when reordering OFS files:

- o create a backup of the old OFS files using script fs5fcopy
- o if any file sizes are to be changed:
 - change the input file for program FILESIZE
 - run program FILESIZE
 - check the program and log output from FILESIZE for errors
 - copy the FILESIZE punch output file to the FILECRAT program input directory
- o run the script ofs_reorder
- o check the program and log output for programs FILECRAT, PRDUTIL and REORDER for errors
- o run the program FCST using the old OFS files and the new OFS files and compare the program and log output
- o copy the new OFS files to the old OFS files using script fs5fcopy

OFS Locks

An OFS lock is a temporary file that indicates when a certain OFS file set is in use. A lock file is opened by a program either with read or write access rights. If a second program tries to open the same lock when a currently running program has already opened it with write access, the second program waits for a certain amount of time and then tries to open the lock again. If the second program cannot open the lock within a default amount of time, it should stop. When a program finishes it should call a routine to free the lock. If a program ends abnormally or does not free the lock and ends, the lock file will revert to a non-use status since UNIX will free the file when a process dies.

The following Apps_default tokens are used to define the parameters needed for the lock:

```
locks_dir ..... directory name to hold lock files
ofs_files ..... path name to directory holding OFS
                  file sets
ofs_level ..... subdirectory under ofs_files that
                  holds the specific file set used by an
                  executable (full path name:
                  <ofs_files>/<ofs_level>/fs5files)
ofs_fs5files ..... used as a check for the same full path
                  name of the file set
ofs_lock_max_wait ..... maximum number of minutes to wait to
                  get a lock before a status flag
                  indicates that the program should be
                  aborted
ofs_lock_wait_interval ... number of seconds between retries to
                  get a lock
```

The following routines are called to use the locking system:

- o set_ofs_lock which opens a lock
- o free_ofs_lock which frees the lock

Routine set_ofs_lock has two arguments, the first is either the character string 'read' or 'write' and the second is the returned status (0 for lock opened, 1 for cannot open because the lock is in use by another program, 3 for cannot open due to a program error).

Routine free_ofs_lock has one argument which is the returned status of the free lock operation (0 for no error, 1 for error).

The routine UPINIO also needs to be called. This routine sets variable UE in block common UPDAIO to the FORTRAN output unit number for status/error messages. It uses the following tokens to set the unit number (set to -1 if no messages are desired):

```
ofs_error_output ..... set to 'on' for output to standard
                        error output, else set to 'off'
fortran_stderr ..... set to the FORTRAN standard error unit
                        number
```

The following is an example of using the lock routines:

```
CHARACTER*5 LTYPE
```

```

...
CALL UPINIO ( )
...
LTYPE = 'write'
CALL SET_OFS_LOCK (LTYPE, ISTAT)
IF (ISTAT.NE.0) STOP 16
...
CALL FREE_OFS_LOCK (ISTAT)
...

```

The following is a list of routines required and the libraries where they can be found

```

free_ofs_lock.f      .../nwsrfs/util/lib/libnow_r.a
ftn_std_err.f       .../nwsrfs/util/lib/libnow_r.a
lockoff.f           .../nwsrfs/util/lib/libnow_r.a
lockon.f            .../nwsrfs/util/lib/libnow_r.a
locks.c             .../nwsrfs/util/lib/libnow_r.a
make_ofs_lock.f     .../nwsrfs/util/lib/libnow_r.a
set_ofs_lock.f      .../nwsrfs/util/lib/libnow_r.a
upinio.f            .../nwsrfs/util/lib/libnow_r.a
get_apps_defaults.c .../nwsrfs/util/lib/libutil_gen1_r.a
kkpos.f             .../nwsrfs/util/lib/libutil_gen1_r.a
datim2.c            .../nwsrfs/util/lib/libdate_time_r.a
uwait.f             .../nwsrfs/util/lib/libdate_time_r.a

```

The lock name is created by first creating two path names and then checking if they are the same:

1. Get apps_defaults values for tokens ofs_files and ofs_level and create the path name <ofs_files>/<ofs_level>/fs5files.

Example:

```

<ofs_files> = /awips/rfc/nwsrfs/ofs/files
<ofs_level> = ofstest
pathname    = /awips/rfc/nwsrfs/ofs/files/ofstest/fs5files

```

2. Get apps_defaults values for token ofs_fs5files as a single complete path name.

Example:

```

pathname = /awips/rfc/nwsrfs/ofs/files/ofstest/fs5files

```

If the path names are the same, the lock name is set to ofs.<ofs_level>.

Example: lockname = ofs.ofstest

If the path names are different (such as an environment variable being reset for one of the tokens in a script), the lock name is set to the value of token ofs_fs5files with the slashes changed to underscores.

Example: lockname = _awips_rfc_nwsrfs_ofs_files_ofstest_fs5files

To use the lock routines but supply a completely different type of

lock name, routine make_ofs_locks could be replaced with a new routine that uses the different type of lock name.

The calling sequence for routine make_ofs_locks is as follows:

```
CALL MAKE_OFS_LOCK (LOCK_NAME,LOCK_LEN,ISTAT)
```

where LOCK_NAME is a CHARACTER*(*) output variable that contains the lock name
LOCK_LEN is an INTEGER*4 output variable that contains the number of characters in LOCK_NAME
ISTAT is an INTEGER*4 output variable that contains the status code (0=no error, 1=error)