TO: MSPM Distribution

FROM: R.M. Graham

DATE: June 17, 1966

Section BE.10.01 (dated 6/10/66) obsoletes section BE.10.00 (dated 5/10/66).

Published: 6/10/66

Identification

Elementary File System (EFS)
D. Levinson, L. B. Ratcliff

Purpose

The Elementary File System (EFS) is a free-standing GE 635 package which provides input/output capability for programs within the 6.36/64.5 systems via an EPL - compiled object-time EPL-I/O module.

Introduction

EFS capabilities are made available to the 6.36/64.5 user by a package of interface routines which escape from 645 simulation, prepare argument lists, and accept arrays which may be non-contiguous (i.e., pages) in simulated 645 core.

The EPL user utilizes EFS through standard EPL I/O statements routed at object-time through the EPL-I/O module. Calls to EFS entries are made by the EPL object-time I/O package. These entries, as stated above, are also available to the user who finds it necessary to perform I/O in assembly language procedures. The major EFS functions (open, write, read and close) are discussed under separate headings.

Each file which is opened during a process is considered an active file. Files are also classified according to their source or destination. A permanent file resides in 6.36/64.5 permanent file storage. A foreign input file is one whose origin (relative to this execution) is external to the system (e.g., CTSS for a 6.36 user). A foreign output file is one whose destination is external to the system. A temporary file is an interim file created during the process. File handling is discussed in more detail under the heading, File utilization.

The user has the option of printing a file as it is created or read (printd), or of printing a file at the time it is closed (printw). These procedures are described in Section BE.10.02.

Lists of EFS restrictions and error codes are also included under separate headings.

The Open Function

A file is opened by the statement,

```
CALL OPNEFS (OPCLAL)
```

whose argument is a structure of the following form (See figure 1):

```
DECLARE 1 OPCLAL,
        2 FNPNTR    POINTER,
        2 FNLNTH    FIXED,
        2 FNOFST    FIXED,
        2 DRCTN     BIT  (36),
        2 SRCESP    BIT  (36),
        2 STATUS    BIT  (36),
        2 IRECNO    FIXED,
        2 IOFFST    FIXED;
```

FNPNTR (File name pointer)

FNLNTH (File name length)
The number of characters in the file name.

FNOFST Not used.

DRCTN (Direction)
If bit 1 is 0, the file is being opened for input. If bit 1 is 1, the file is being opened for output.

SRCDSP (Source-disposal option)
For input files:
Bit 1 = 1 if foreign file (Bit 2 must be zero.)
Bit 2 = 1 if permanent file (Bit 1 must be zero.)
For output files:
Bit 1 = 1 indicates the file is to be placed on the foreign file output tape at the end of the process.
Bit 2 = 1 indicates the file is to become permanent at the end of the process.

STATUS (Status return information)

Upon return,
Bit 1 = 1 indicates that some error or inconsistency has been encountered in attempting to open the file. A two-character error code appears in bits 19 through 36.

IRECNO (Logical record number)
Upon return, contains the number of the last logical record of the file.
The file is empty if IRECNO contains one and IOFFST contains zero.

IOFFST (Offset)
Upon return, contains the number of words in the last logical record of the file, or is zero if the file is empty.

IRECNO and IOFFST constitute a "write pointer" which points to the next sequential word of the last appended logical record (for a non-empty file) or to the first word, (word zero), of logical record one (for an empty file).

If IRECNO contains the value one and IOFFST contains a non-zero value, that value is the number of words in the file. If IRECNO contains the value one and IOFFST contains zero, the file is empty. If IRECNO is greater than 1, it specifies the number of records in the file, and IOFFST the number of words in the last record.

If, after opening, the user issues a call to write with record number set equal to IRECNO and an offset equal to IOFFST, the write will append to the last record.

The Write Function

Logical records are written or augmented by the statement

```
CALL WRTEFS (RDWTAL)
```

whose argument is a structure of the following form (see figure 2):

```
DECLARE 1 RDWTAL,
        2 FNPNTR   POINTER,
        2 FNLNTH   FIXED,
        2 FNOFST   FIXED,
        2 LOCFS    POINTER,
        2 RECNUM   FIXED,
        2 OFFSET   FIXED,
        2 NUMWDS   FIXED,
        2 NUMTST   FIXED,
        2 STATUS   BIT (36);
```

FNPNTR (File name pointer)
Same as in open.

FNLNTH (File name length)
The number of characters in the file name.

FNOFST Not used

LOCFS (Array pointer)
Points to the beginning of the array to be output.

RECNUM (Logical record number)
The normal mode of writing requires the user to specify a logical record number and a word offset within that record. In this case, RECNUM must contain the number or the number +1 of the last logical record written. Record number 1 is the first logical record of each file. However, if RECNUM contains zero, an alternate mode of writing a file sequentially is indicated.

When writing in this "sequential" mode, the user sets bit 2 of the status word (STATUS). If bit 2 = 1, the current logical record is appended and then terminated. A subsequent write will begin a new record. If bit 2 = 0, the current logical record is appended but not terminated. The next write will also append this record.

The user must be aware that bit 2 of STATUS is interrogated only if OFFSET (below) is also zero. If OFFSET is non-zero, an attempt is made to append the current logical record.

OFFSET (Offset)
OFFSET contains a value n, which is the size of the specified logical record number. The first word of the array (LOCFS) is output as the (n + 1)th word of the specified logical record. Word zero is the first word of each logical record. If a new logical record is being written, OFFSET must contain zero. Non-zero values in OFFSET are used when appending words to a logical record.

NUMWDS (Number of words to be written)

NUMTST (Number of words transmitted)
This value, returned to the user, is the number of words actually written (NUMTST = NUMWDS unless STATUS, below, indicates error).

STATUS (Status return information)
Upon return, bit 1 = 1 indicates that some error or inconsistency has been encountered in attempting to write the record. A two-character error code appears in bits 19 through 36.

When writing in the sequential mode (RECNUM contains 0) the user sets bit 2 of STATUS as input information. If bit 2 = 1, the current logical record is terminated after writing. A subsequent write will append to the current record.

The Read Function

Files are read using the statement

CALL READFS (ARGLST)

with the argument as described for WRTEFS (See figure 2).

FNPNTR (File name pointer)

FNLNTH (File name length)
The number of characters in the file name.

FNOFST Not used.

LOCFS (Array Pointer)
Points to the beginning of the array into which data is to be read.

RECNUM (Logical record number)
If the value in RECNUM is non-zero, it specifies the logical record number to be read.

If RECNUM contains zero, the sequential mode of reading is assumed. When reading in this mode OFFSET must contain zero. The file is always positioned so that reading begins with the first word following the last word transmitted via the previous read command. Since logical record boundaries are not crossed by a single command, if the user requests 30 words to be read and the file is positioned at the 10th word of a 20-word logical record, 11 words are actually transmitted, the value 11 is returned to the user in NUMTST, bit 2 of the status word is set to one, and the file is positioned at the first word of the next sequential logical record.

OFFSET (Offset)
The word number (within the specified logical record) of the first word to be transmitted. (OFFSET must contain zero if RECNUM contains zero, indicating next sequential word is to be read).

If RECNUM is non-zero and OFFSET contains zero, then the logical record specified by RECNUM is read beginning with the first word of the record. If OFFSET contains a non-zero value, n, then data transmission begins with the (n +1)th word of the specified record. Reading may be random; however, logical record boundaries cannot be crossed. When the last word read completes the specific read request and is the last word of

the logical record, an indicator is returned to the user in bit 3 of the status word. If an end of logical record is encountered, reading terminates and an end of record indicator is returned to the user in bit 2 of the status word.

NUMWDS (Number of words to be read)
Number of words the user is requesting.

NUMTST (Number of words transmitted)
The number of words actually read. (Value returned to the user.)

STATUS Upon return, bit 1 = 1 indicates that some error or inconsistency has been encountered in attempting to read the record. A two-character error code appears in bits 19 through 36. Bit 2 = 1 indicates an end of record has been encountered in attempting to read additional words. In this event, the number of words actually transmitted is less than the number requested. Bit 3 = 1 indicates no more words are in the specified logical record; that is, the last word read was the last word of the logical record. Bit 4 = 1 indicates an end of file has been encountered.

The Close Function

A file is closed by the statement

```
CALL CLSEFS (OPCLAL)
```

whose argument is a structure of the following form (see figure 1):

```
DECLARE 1 OPCLAL,
        2 FNPNTR   POINTER,
        2 FNLNTH   FIXED,
        2 FNOFST   FIXED,
        2 DRCTN    BIT (36),
        2 SRCDSP   BIT (36),
        2 STATUS   BIT (36);
```

FNPNTR (File name pointer)

FNLNTH (File name length)
The number of characters in the file name.

FNOFST Not used.

DRCTN The contents of DRCTN are not interrogated by CLSEFS.

SRCDSP (Source-disposal option)
Bit 1 = 1 indicates the file is to be placed on the foreign file output tape at the end of process.

Bit 2 = 1 indicates the file is to become permanent at the end of process.

Bit 3 = 1 indicates the file with this name in permanent file storage is to be deleted at the end of process.

STATUS (Status return information)
Upon return, bit 1 = 1 indicates that some error or inconsistency has been encountered in attempting to close the file. A two-character error code appears in bits 19 through 36.

File Utilization

Once a file has been opened during a process, it is one of the ten (or fewer) active files and remains so for the duration of the process regardless of its mode (open or closed). Any file which has been closed may subsequently be reopened. When a file is opened, it must be opened specifically for input or for output. The I/O mode is altered by closing a file and then reopening it in the alternate mode.

It should be noted that, during a process, once a file has been opened, its source (Permanent or foreign) need not be repeated on subsequent reopening of the same file. During the process, an active file is treated without regard to its original source or to its ultimate destination (or destinations). The functions of deleting files from permanent file storage and of moving an active file to permanent file storage and/or to the foreign file output tape are performed only at the end of the process.

A file is temporary by default. That is, if it is created during the process but not directed to permanent or foreign status at any time via the open or close argument list, then it is temporary.

A file open for output is written sequentially. The file may be augmented by the next sequential logical record; also, the last logical record written may be extended beginning with the next sequential logical word.

A file open for input may be read sequentially or randomly.

During a process on the 6.36/64.5 system, if the user wishes to read or augment an existing file he must specify whether the file is to be obtained from

permanent file storage or from the foreign file input tape. (Foreign files are made available to EFS by the user via the Merge-editor or 64.5 Driver). The user may also direct EFS to place an active file in permanent file storage or on the foreign file output tape (or both) at the end of process. Source is specified when initially opening the file, and destination may be specified when opening or closing the file.

The following chart indicates the resultant action for the possible open/close option combinations.

	File already exists as:			File does not exist:			Delete
	Perm.	Frgn.	Temp.	Perm.	Frgn.	Temp.	
Action for the initial open and file is to be read	1	2	N	E	E	E	N
Action for the initial open and file is to be written	3	4	N	5	6	7	N
Action for close	8	9	10	E	E	E	11

N - This combination of options cannot occur.

E - This combination is not allowed. Bit 1 of the user status word is set to 1 and control is returned to the user. (See description of open and close.)

Action Number	Action
1	The permanent file is made active.
2	The foreign input file is made active.
3	The permanent file is made active. At the end of process the active file replaces the original permanent file in permanent file storage.
4	The foreign input file is made active. At the end of process the active file is placed on the foreign file output tape.
5	An active file with the specified name is created. At the end of process the active file is placed in permanent file storage.

Action Number	Action
6	An active file with the specified name is created. At the end of process, the active file is placed on the foreign file output tape.
7	An active file with the specified name is created. It ceases to exist at the end of process if not subsequently designated as permanent or foreign output.
8	The active file is placed in permanent file storage at the end of process.
9	The active file is placed on the foreign file output tape at the end of process.
10	Unless previously or subsequently designated as foreign or permanent output, the file ceases to exist at the end of process.
11	The delete option is meaningful only if the file source was permanent file storage and it has been opened only for reading. The permanent file is deleted at the end of process.

The user may direct an active file to permanent and/or foreign status by subsequently opening it for output with the permanent and/or foreign option. However, when an active file is opened for input, a source (permanent file storage or foreign file input tape) need not be specified. If a source is specified it must agree with the source specified at the time the file was made active.

As implied in the preceding paragraph, it is possible for an active file to become both permanent and foreign output at the end of process.

Example 1. Open, permanent for output. Close as foreign.

If the file exists in permanent file storage, it is made active. Otherwise, an active file is created. At the end of process, the augmented (or newly created) file is placed on the foreign file output tape and is placed in permanent file storage (replacing the original if there is one).

Example 2. Open for output. Close as permanent and foreign.

An active file is created. At the end of process it becomes both permanent and foreign as in example 1.

Example 3. Open, permanent for input. Close as foreign.

The permanent file is made active. At the end of process, it is placed on the foreign file output tape.

Example 4. Open, permanent for input. Close as foreign and permanent.

Result is same as in example 3 except that EFS actually deletes the file from permanent file storage and then reinserts it. Example 3 avoids this spurious effort.

Example 5. Open, foreign for input. Close as foreign and permanent.

Here we assume the file exists on the foreign file input tape. At the end of process, the augmented active file is placed in permanent file storage and the foreign file output tape.

Restrictions

The restrictions listed below reflect the initial implementation of EFS.

1. The maximum number of files a user may access during a process is ten.

2. The maximum number of logical records for all active files combined is 1000. (This limit is actually imposed by the size of the EFS disc table, all of which must be retained in core.) The limit may alternately be stated as a maximum of 320,000 words. These are "best possible" maximums; actual limits indeed may be lower. In EFS, each logical record consumes at least 320 words of disc storage and at least one entry in the EFS disc table. Logical records of fewer than 320 words reduce the maximum limit on the number of words; logical records of more than 320 words reduce the limit on the number of logical records. Very short logical records make very inefficient use of EFS.

3. Approximately three million words of disc are available for permanent file storage.

4. The maximum number of characters in a file name is 32.

5. When a file is open, it is open for reading or for writing, not both.

6. A file must be closed in order to assure all its information has actually been written.

7. No single read or write command may attempt to transmit more than 64 pages of information. (Page size may be either 64 or 1024.)

8. No error recovery procedure is available to the user other than his own construction.

9. There is no access control for permanent files.

Error Codes

(To be included at a later date.)

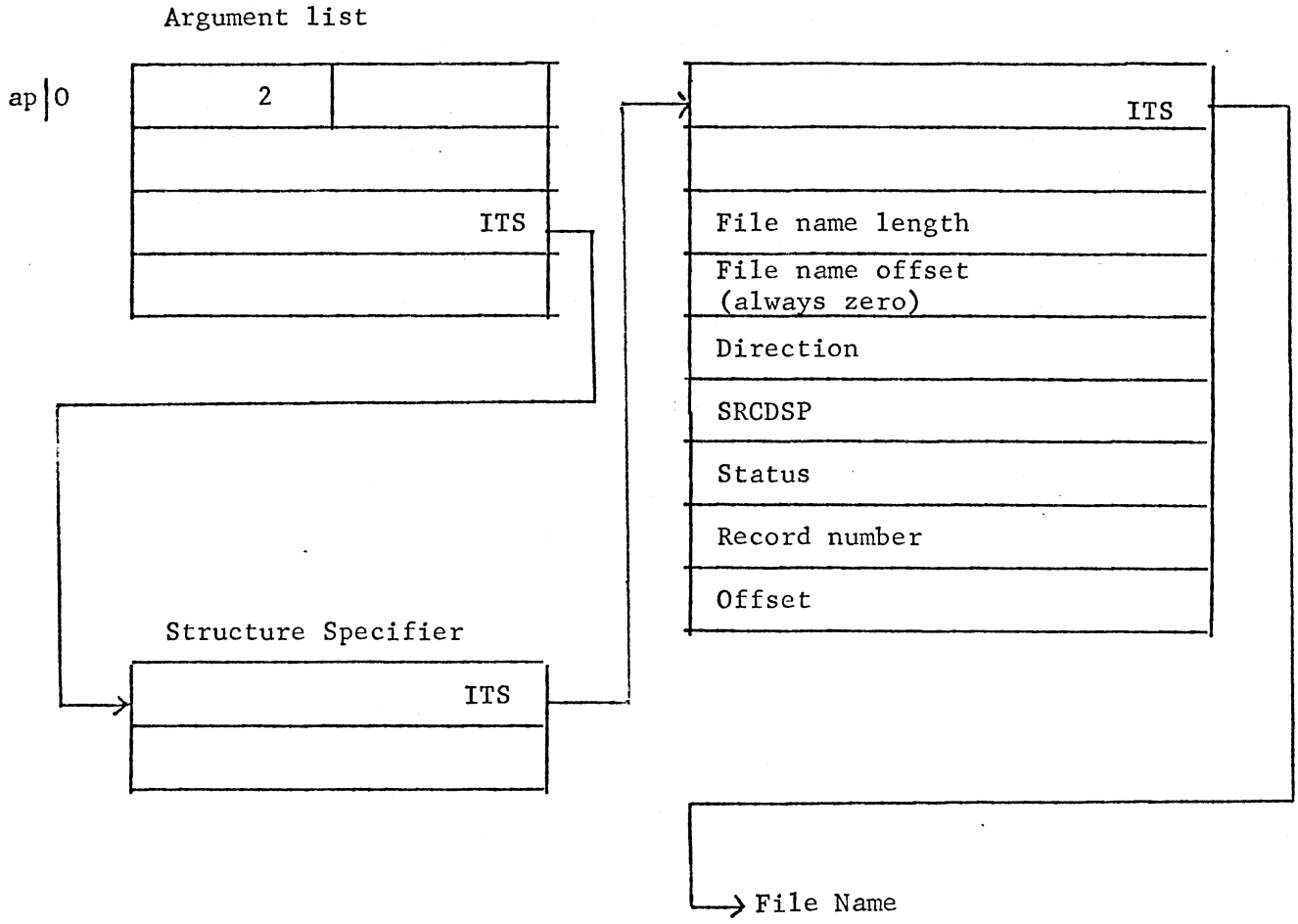

Figure 1 - Open/Close Argument List Structure
(Portions of the structure not used by EFS are not shown.)

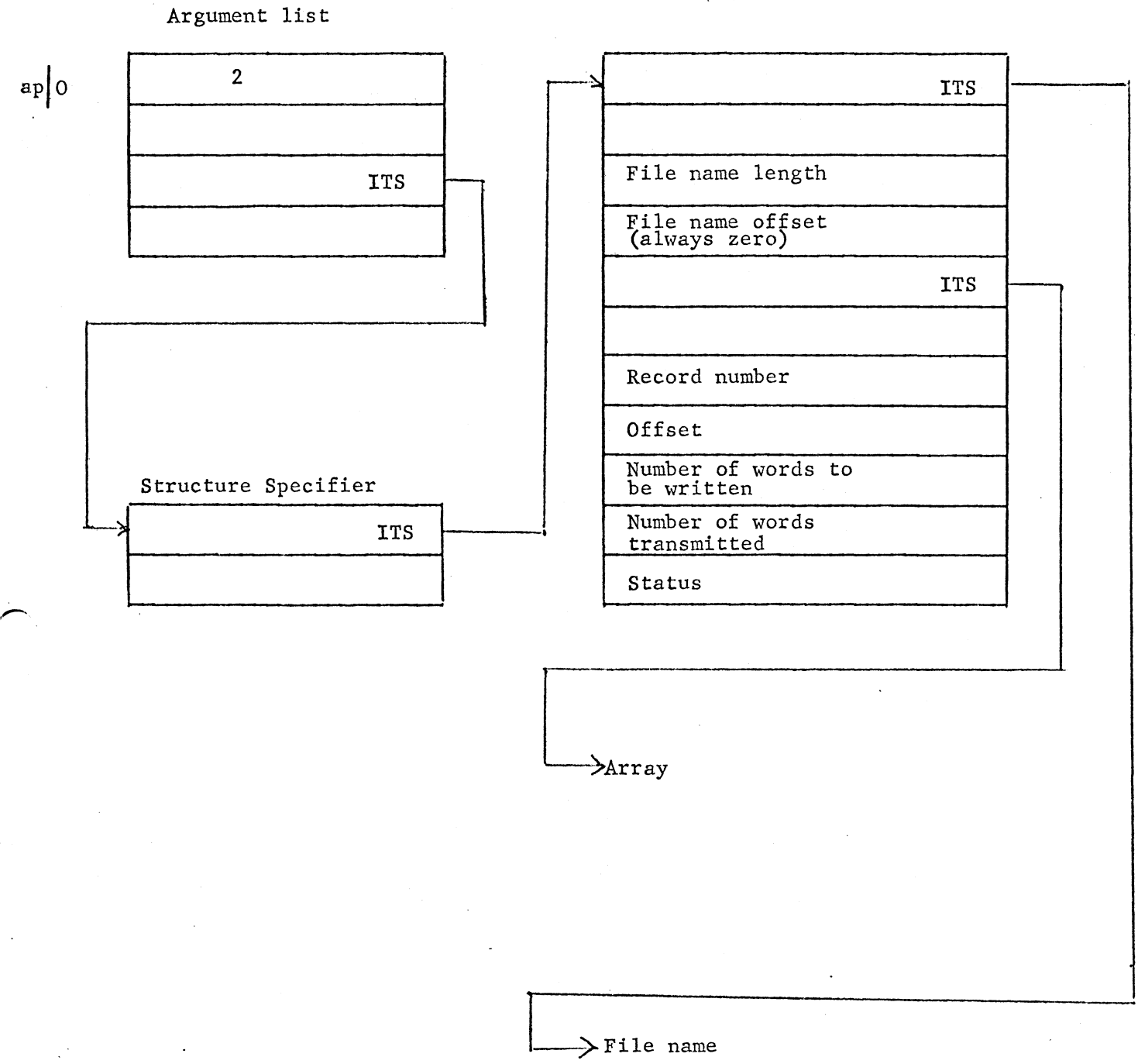

Figure 2 - Read/Write Argument List Structure
(Portions of the structure not used by EFS are not shown.)