3 System architecture

CHAPTER ~~4 -- SYSTEM CALLS~~

This chapter needs an introduction.

3.1

~~4.1~~ USER-SYSTEM INTERACTION

The CAL Time Sharing System provides a number of actions which are available to the user so that he can interact with the system. The actions apply to the objects created and maintained by the ECS and disk systems: files, maps, allocation blocks, event channels, capability directories lists (C-lists), operations, processes, and class codes. A record is kept in a table in ECS, called the Master Object Table (MOT), of all objects existing in ECS at any given time. Each entry in MOT gives the name of the object and its ECS location.

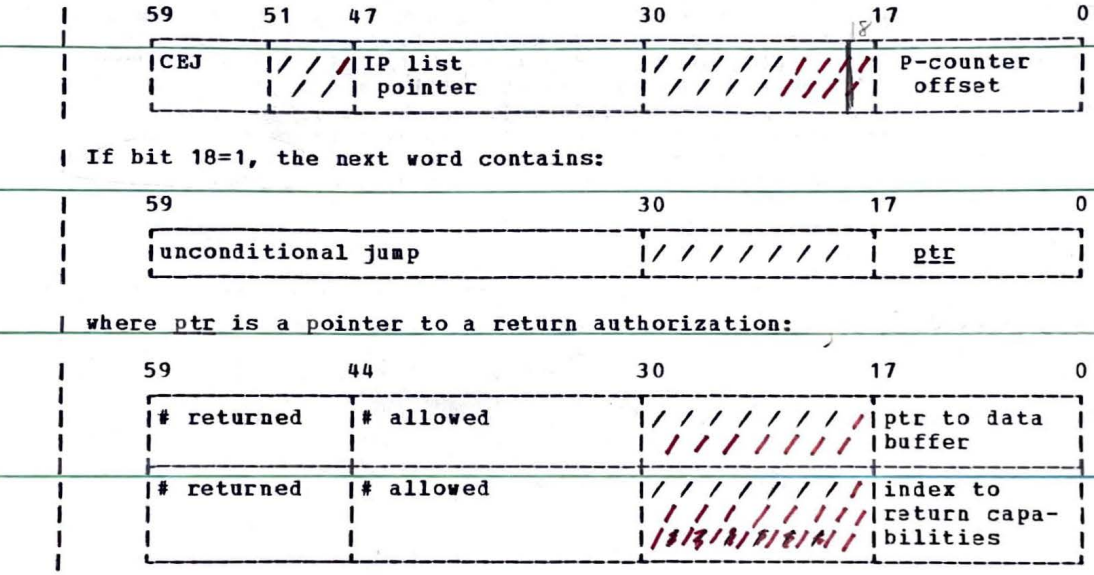
fields

The user makes a call upon the system by setting up the appropriate parameter list for the action he wants to initiate, prior to passing control to the system entry/exit routines by executing a CEJ instruction. (The CEJ, Central Exchange Jump, causes the current contents of the 6400 central processor's registers to be exchanged with a similar 16 word package in Central Memory.) The system entry/exit routines determine the nature of the user's call, collect and check the parameters needed for the action, transfer control to the proper system action routine, and finally, return control to the user (by another CEJ, which restores the registers) after the system action is completed.

4.1.1 Requesting a System Action

The CEJ instruction used to call the system supplies the information required to initiate the action and return to the user. (See Figure 1.) In particular, it is expected that the CEJ was in the upper 30 bits of the instruction word; of these 30 bits, the lower 18 bits are used by the system to locate the user's input parameter (IP) list. If this 18 bit field is negative, the complement of the low order 4 bits specify which register in the user's exchange package contains the input parameter list pointer (e.g., -3 -> B3; -10 -> X2). Otherwise, the 18 bit field itself is taken to be the IP list pointer. This pointer is checked for legality (i.e., it must be positive and less than the user's field length) and an error is generated if appropriate.

Figure 1. System Call Instruction



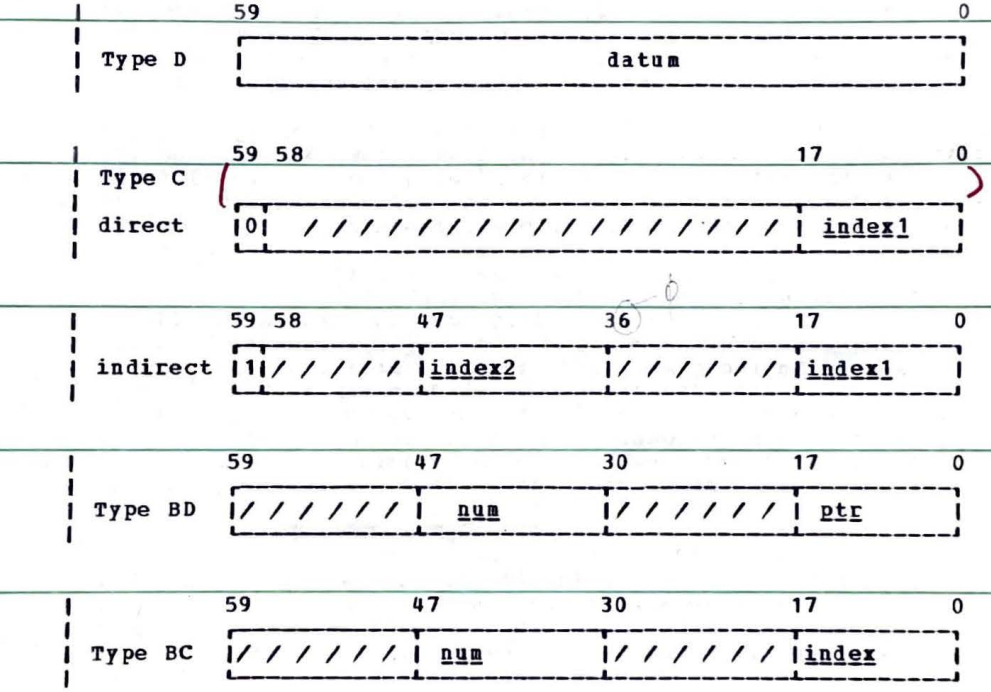
can we indicate bit 18 separately?

001 N + | Each entry in the input parameter (IP) list may be one of four types
002 + | (see Figure 2): type D, a 60-bit data item; type C, a capability
003 + | specifier; type BD, a block data specifier; or type BC, a block
004 + | capability specifier. Type C, a capability specifier, may be either
005 + | direct or indirect. The direct form specifies the capability at index
006 + | 1 in the user's capability list (local C-list), while the indirect form
007 + | requires that index1 be the index in the full C-list of a capability
008 + | for a C-list; the specified capability is at index2 in the C-list
009 + | given by index1. Each capability residing in a C-list authorizes
010 + | access to a particular object in addition to giving the object's type
011 + | (file, process, event channel, etc.) and the set of actions allowed on
012 + | that particular object (option bits). See section 3.7 for a full discussion of capabilities + C-lists

delete space

full

Figure 2. Input Parameter (IP) List Types



001 N +
S
002 S1 + The first parameter, that is, the first word of the IP list (called
003 + IP0), is always expected to be an index into the user's capability
004 + list. This parameter, after being checked for legality (i.e., it must
005 + be positive and within the range of the full C-list), is used to fetch
006 + the capability for the operation which specifies the action to be
007 + performed, and the nature of the parameters of the action. (If the
008 + capability is not for an operation, an error is generated.)
S
009 S1 + Operations are ECS objects which direct the transfer of control from
010 + the user to the system when the user calls upon the system. They
011 + identify the action(s) to be taken by the system and direct the passing
012 + of parameters to the system or between user subprocesses (see 2.3
013 + SUBPROCESSES). An operation consists of one or more orders, each of
014 + which designates a particular system action, and a set of parameter
015 + specifications which indicate the type (capability or datum) of each
016 + parameter required for the action and the required options for each
017 + capability parameter. Basically, the parameter specifications in the
018 + operation are of two genere -- parameters which are permanently "fixed"
019 + within the operation and those that are provided by the user. When an
020 + operation is first created, before any of its parameters have been
021 + specified, all parameter specifications are typed "none". Before the
022 + operation can be used, all of its parameters must be specified using
023 + the actions provided for specifying operation parameters. If the
024 + action is parameterless, the operation contains no parameter
025 + information.
S
026 S1 + The system entry/exit routine reads the first order of the operation
027 + and uses the parameter specifications to construct an actual parameter
028 + (AP) list. This list consists of parameters which are "fixed" in the
029 + operation and of user-supplied parameters drawn from the IP list. The
030 + IP list should contain, in successive words, datum parameters, (type D)
031 + which are transferred directly to the actual parameter list; C-list
032 + indices (type C), which designate capabilities in the user's full
033 + C-list; block data parameters (type BD), which specify a block of data
034 + words to be transferred to the actual parameter list; and block
035 + capability indices (type BC), which specify a number of capabilities to
036 + be transferred. Two words are copied to the actual parameter list for
037 + each capability parameter (capabilities are two words long) and one
038 + word is copied for each datum parameter. During the construction of
039 + the actual parameter list, errors will be generated if 1) a C-list
040 + index is bad (i.e., is negative or outside the full C-list); 2) if the
041 + type and options (indicated below by "OB.x") in the capability do not
042 + correspond to those specified by the parameter information in the
043 + operation (this checking is not performed if the parameter specifica-
044 + tion is "any capability"); or 3) if a "none" parameter specification is
045 + encountered in which case parameter processing terminates.

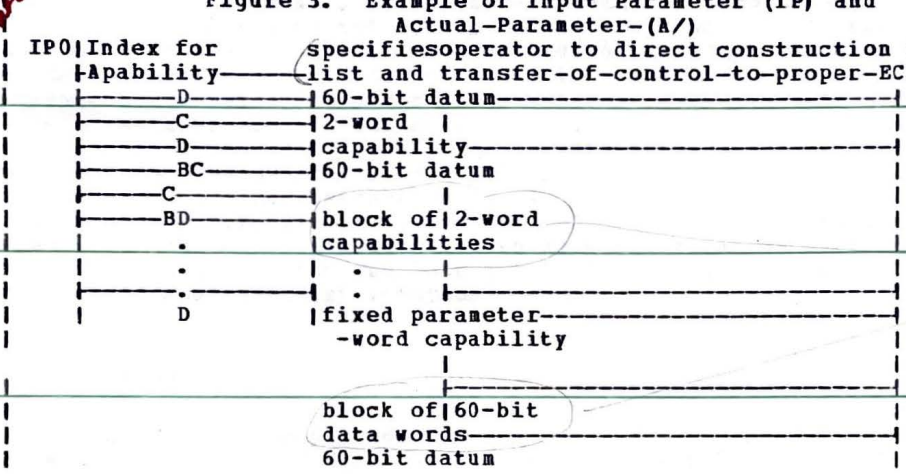


Figure 3. Example of Input Parameter (IP) and Actual-Parameter-(A/)

After the actual parameter (AP) list is completed, the operation is checked to see if the action is a subprocess call or jump. If so, a flag bit will indicate the presence of a class code (the subprocess name) in the operation. In this case, the operation also contains a parameter type bit mask indicating the type (capability or datum) of each parameter. The system entry/exit routine places the class code from the operation, the number of parameters, and the bit mask into the user's process descriptor in the actual parameter list area.

Finally, the ECS action number is extracted from the operation and is used as an index to a jump to the proper entry point for the desired ECS action. When the action is completed, control returns to the ~~user~~ caller by one of the mechanisms described below.

F-returns

Under some conditions, when the normal function of an ECS system action cannot be carried out but the condition is not serious enough to warrant the generation of an error, an F-return will result. If this occurs, the count of F-returns initiated for the operation is increased, and the operation is checked to see if it contains any more orders (which are specified as alternative actions). If so, the next order of the operation is interpreted. This process is identical to the one described above, except that the actual parameter list contains the parameters for all orders up to and including the current one. If the F-return count reaches the number of orders in the operation, (i.e., all alternative actions have been tried and have ended in an F-return) control is returned to the user.

There are two different ways in which control is returned to the user depending upon whether an action completed normally (possibly after one

Normal return

F-return

Return + redo

No return at all

or more F-returns) or the F-return count became equal to the number of orders in the operation. The normal return causes the user's P-counter to be incremented by the number supplied by the user in the low order 18 bits of the CEJ instruction word originally used to call the system. The new P-counter must be positive and less than the user's field length; otherwise, an error is generated. When the return to the user results from an ultimate F-return, the user's P-counter is left unchanged.

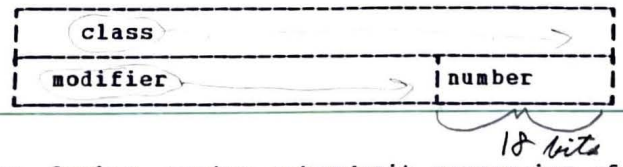
Some actions attempt to return parameters to the caller. When this happens, as in the case of the 'return with parameter' operation (see 4.5 ...),

Bit 18 of the CEJ instruction is also checked. If zero, no return of data or capabilities is authorized. If one, the word following the CEJ must contain a pointer to a return authorization as shown in figure 1. The # returned fields are both set by the system to the actual number of data (capabilities) returned.

4.1.2 Errors:

The use of improper parameters in making an ECS system call is considered to be an error on the part of the process which is making the call. When an error is detected, it is first assigned an error class and number. The class identifies the type of the error, while the number pinpoints the particular error within a type. Furthermore, associated with each subprocess within a process is an error selection mask (ESM) indicating the classes of errors the subprocess is prepared to handle. The "ancestors" of the current subprocess (see ?) are checked (starting with the current subprocess) to find a subprocess whose ESM indicates it is willing to handle this class of errors. The subprocess which accepts the error is called and is passed the error class and number as AP1 and AP2. (See Figure 4.) Execution in the error processing subprocess is initiated at the normal entry point-1. A precaution is taken against error loops; the subprocess which accepts the error is temporarily disqualified from accepting any more occurrences of the errors in the same class.

Figure 4. Error Class and Number



Possible errors during system entry/exit processing of an ECS system action call:

~~1 Used when the return is via a "Return with parameter" operation (see 4.5 Subprocess Call Actions).~~

		Class	Number	Error Description
001	S1 T T+ S			
002	S1 T T+	E.PARMS	E.NEGPT	The IP list pointer address is <- 20
003	S0 T T+	E.PARMS	E.BIGPT	The IP list pointer address is greater than the
004	S0T +			user's field length
005	S0 T T+	E.PARMS	E.NEXIX	C-list negative
006	S0 T T+	E.PARMS	E.BIGIX	C-list index too large (not within full C-list)
007	S0 T T+	E.OPER	E.IPO	First parameter (IPO) does not point to a capability
008	S0T +			for an operation
009	S0 T T+	E.OPER	E.NOOP	The operation does not exist
010	S0 T T+	E.OPER	E.PSANY	"ANY" parameters specification encountered
011	S0 T T+	E.OPER	E.CAPTY	Type or options bad for a capability parameter
012	S0 T T+	E.MISCE	E.CLMOT	C-list does not exist
013	S0 T T+	E.OPER	E.MANPAR	IP list extends past user's field length
014	S0 T T+	E.PARMS	E.NEGPAR	The new P-counter is negative on return to user
015	S0 T T+	E.PARMS	E.BIGPAR	The new P-counter exceeds the user's field length
016	S0 T T+	E.NOERR	E.NOERR1	No subprocess to take error class

```
PAG LIN                    CODE

001 001 +FRONT+
001 002 +SWIDTH71+
001 003 +DEPTH50+
001 004 +LENGTH71+
001 005 +MARGIN0+
001 006 +TITLE+SYSTEM CALLS
001 007 +SUBTITLE+January 1971
002 003 +SETTAB2=13,3=19,4=39,5=55,6=69+
002 004 -S2AT1-59-T2-51-T3-47-T4-30-T5-17-T6-0
002 005 -S0B5,13,19,39,55,69-
002 006 -S0T1- CEJ                -T2- / /-T3- IP list -T4- / / / / / -T5-  P-counter
002 007 -S0T2-  / / -T3-  pointer-T4-  / / / / -T5-   offset
002 010 -S1AT1-59-T4-30-T5-17-T6-0                                                        -T5-
002 011 -S0B5,39,55,69-
002 012 -S0T1- unconditional jump-T4- / / / / / / /
002 015 -S1AT1-59-T3-44-T4-30-T5-17-T6-0                                                  -T5- pt
002 016 -S0B5,19,39,55,69-
002 017 -S0T1- # returned -T3- # allowed -T4- / / / / / / /
002 018 -S0T5- buffer
002 020 -S0T1- # returned -T3- # allowed -T4- / / / / / / / -T5- index to
002 021 -S0T5- return capa-
002 022 -S0T4-  / / / / / / -T5- bilities
003 013 +SETTAB+
004 003 +SETTAB1=10,2=23,3=36,4=49,5=60+
004 004 -S3T1-59-T5-0
004 005 -S0B10,60-
004 006 -S0-Type D -T3-datum
004 008 -S2T1-59 58 -T4-17-T5-0
004 010 -S0B10,12,49,60-
004 011 -S0-direct-T1-  0   / / / / / / / / / / / / / / / / / / / -T4-  index1
004 013 -S2T1-59 58 -T2-47-T3-36-T4-17-T5-0
004 014 -S0B10,12,23,36,49,60-
004 015 -S0-indirect -T1- 1 / / / / /      -T2- index2  -T3- / / / / / / -T4- index1
004 017 -S2T1-59-T2-47-T3-30-T4-17-T5-0
004 018 -S0B10,23,36,49,60-
004 019 -S0-Type BD -T1- / / / / / /      -T2-   num -T3- / / / / / / -T4-  ptr
004 021 -S2- -T1-59-T2-47-T3-30-T4-17-T5-0
004 022 -S0B10,23,36,49,60-
004 023 -S0-Type BC -T1- / / / / / / -T2-  num -T3- / / / / / /     -T4- index
004 025 +SETTAB+
006 003 -S0C-Actual Parameter (AP) List Interactions -G4,20--T4-/<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

        T360.162  ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES
006 003 -S0C-Actual Parameter (AP) List Interactions -G4,20--T4-/<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>-T4-specifiesoperator to direct construction of AP
006 004 -S0-IP0 -V4,20,29--T1-Index for  transfer of control to proper ECS action code. -34,20,29,60- -VV60-
006 005 -AT1-Apability -T4-list and
```

T360.167 ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING VERTICAL LINE

006 005 -AT1-Apability -T4-list and transfer of control to proper ECS action code. -G4,20,29,60- -VV60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

T360.184 AS-IS TEXT EXCEEDS COLUMN WIDTH

006 005 -AT1-Apability -T4-list and transfer of control to proper ECS action code. -G4,20,29,60- -VV60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 006 -AT2- D -T4- 60-bit datum -G4,20,29,60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

T360.162 ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES

006 006 -AT2- D -T4- 60-bit datum -G4,20,29,60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 007 -AT2- C -T4- 2-word -G4,20-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

T360.162 ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES

006 007 -AT2- C -T4- 2-word -G4,20-

```
PAG LIN                    CODE

<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 008 -AT2- D -T4- capability -G4,20,29,60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

          T360.162  ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES
006 008 -AT2- D -T4- capability -G4,20,29,60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 009 -AT2- BC -T4- 60-bit datum -G4,20-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

          T360.162  ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES
006 009 -AT2- BC -T4- 60-bit datum -G4,20-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 010 -AT2-C -G4,20-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

          T360.162  ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES
006 010 -AT2-C -G4,20-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 011 -AT2-BD -T4- block of 2-word -G4,20-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

          T360.162  ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES
006 011 -AT2-BD -T4- block of 2-word -G4,20-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 012 -AT2- . -T4- capabilities
006 013 -AT2- . -T4-     .
006 014 -AT2- . -T4-     . -G4,20,29,60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

          T360.162  ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES
006 014 -AT2- . -T4-     . -G4,20,29,60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 015 -AT2- D -T4- fixed parameter -Q4,20- -G29,60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

          T360.162  ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES
006 015 -AT2- D -T4- fixed parameter -Q4,20- -G29,60-
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 016 -AT4-  -word capability
006 017 -S1G29,60-
006 018 -AT4- block of 60-bit
006 019 -AT4- data words -G29,60
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

          T360.162  ATTEMPT TO OVERLAY VALID CHARACTER WHILE DRAWING HORIZONTAL LINES
006 019 -AT4- data words -G29,60
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

          T360.182  INVALID EDIT CODE
006 019 -AT4- data words -G29,60
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
```

```
        T360.183  EDIT CODE PROCESSING OUT OF SYNCHRONIZATION
006 019 -AT4- data words -G29,60
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
006 020 -AT4- 60-bit datum -Q-
007 011 must contain a pointer to a return authorization¹ as shown in figure 1.
007 012 -F-¹ Used when the return is via a "Return with parameter" operation (see
007 013 4.5 Subprocess Call Actions). -Z-
007 014
007 035 +SETTAB1=10,2=16,3=45,4=57+
007 036 -SOB16,57-
007 037 -SOT1-AP1 -T2-     class
007 038 -SOB16,45,57
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

        T360.182  INVALID EDIT CODE
007 038 -SOB16,45,57
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

        T360.183  EDIT CODE PROCESSING OUT OF SYNCHRONIZATION
007 038 -SOB16,45,57
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
007 039 -SOT1-AP2 -T2-   modifier -T3- number
007 041 +SETTAB+
```

```
PAG LIN                    CODE

008 001 -S1-Class -T2-Number -T4-Error Description
008 002 -S1-E.PARMS -T2-E.NEGPT -T4-The IP list pointer address is <- 20
008 003 -S0-E.PARMS -T2-E.BIGPT -T4-The IP list pointer address is greater than the
008 004 -S0T4- user's field length
008 005 -S0-E.PARMS -T2-E.NEXIX -T4-C-list negative
008 006 -S0-E.PARMS -T2-E.BIGIX -T4-C-list index too large (not within full C-list)
008 007 -S0-E.OPER -T2-E.IPO -T4-First parameter (IPO) does not point to a capability
008 008 -S0T4- for an operation
008 009 -S0-E.OPER -T2-E.NOOP -T4-The operation does not exist
008 010 -S0-E.OPER -T2-E.PSANY -T4-"ANY" parameters specification encountered
008 011 -S0-E.OPER -T2-E.CAPTY -T4-Type or options bad for a capability parameter
008 012 -S0-E.MISCE -T2-E.CLMOT -T4-C-list  does not exist
008 013 -S0-E.OPER -T2-E.MANPAR -T4-IP list extends past user's field length
008 014 -S0-E.PARMS -T2-E.NEGPAR -T4-The new P-counter is negative on return to user
008 015 -S0-E.PARMS -T2-E.BIGPAR -T4-The new P-counter exceeds the user's field length
008 016 -S0-E.NOERR -T2-E.NOERR1 -T4-No subprocess to take error class
```

****

File Actions

The Time-Sharing System provides a number of actions for handling files. A description of files and how they are handled under TSS appears in Section 2.1. The following list summarizes the available actions:

ECS File Actions	Disk File Actions
Create a file	Create a file
Create a data block	Create a data block
Delete a data block	Delete a data block
Delete a file	Delete a file
Check missing block (probe)	Check for missing block on disk file (probe)
Read file shape	Open read only (read/write)
Read/write information	Close
Move a data block	Pseudo-close
Test and reset dirty bit	Attach file blocks
	Detach file blocks
	Interchange file contents (shazam)
	Make shared claim
	Make exclusive claim
	Release claim
	Display file information
	Privileged create
	Close all open files ~~for user~~

insert here: Create Disk File via directory system

```
001 S0    +  |                                  Close all open files over-ride
S
002 S0    +  |                                  Return  and  reset  disk  subsystem
S
003       +  |                                  clocks
S
S         •
S
005 S1    +  | Detailed descriptions of these actions appear  below.  For  a  general
S
006       +  | description of how the user calls the system to initiate an action, see
S
007       +  | Section 3.?.
S
S
008 S1    +  | Create an ECS file
S
S
009 S1T   +  |          IP1 C: Capability for allocation block (OB.CRFIL)
S
010 S0T   +  |          IP2 D: C-list index to return capability
S
011 S0T   +  |          IP3 D: Number of levels in the file
S
012 S0T   +  |          IP4 D: Pointer to a list of shape numbers
S
S
013 S1    +  | When  a  file is created, only the file descriptor is constructed.  The
S
014       +  | file descriptor contains a  pointer  to  the  root  of  the  file  tree
S
015       +  | (initially  empty  since  no  data  or pointer blocks exist).  The user
S
016       +  | supplies the capability of the allocation block which is  to  fund  the
S
017       +  | ECS space occupied by the file.  The user must also supply the index of
S
018       +  | the  C-list slot where the system will put the capability for the newly
S
019       +  | created file (all option bits in the capability for the  new  file  are
S
020       +  | turned  on).   The  last  two  parameters  of  the file create action,
S
021       +  | indicate the number of levels (n) contained in  the  structure  of  the
S
022       +  | file  tree, and a pointer to a list of n shape numbers (S1 through Sn),
S
023       +  | the first n-1 of which  indicate  the  number  of  branches  from  each
S
024       +  | pointer block at each successive level of the file tree; the last shape
```

-N-

(EC:CFIL) - These are all the same length so just say for example -T12-(EC:CFIL)

```
001       +   | number  (Sn)  gives the uniform size of all data blocks in the file.  A
S
002       +   | "one level file" (IP3=1) consists of a single data block of length  S1.
S
003       +   | Each shape number (S1 excepted) must be a non-negative power of two.
```

no new page here

Possible errors while creating an ECS file:

Class	Number	Description
E.ABLOCK	E.NOABLK	Allocation block does not exist
E.ABLOCK	E.NOECS	No ECS available
E.PARMS	E.NEGIX	C-list index is negative.
E.PARMS	E.BIGIX	C-list index exceeds full C-list
E.PARMS	E.NEGPT	Pointer to list of shape numbers is negative
E.PARMS	E.NEGPAR	Level number n < 1
E.PARMS	E.BIGPAR	Level number is too large or Pointer to list of shape numbers plus list length exceeds user's FL
E.OPER	E.CAPTY	Type or options bad
E.FILES	E.NEGSIZ	Non-positive shape number
E.FILES	E.BIGSIZ	Shape numbers exceeds $2^{17}-1$
E.FILES	E.NOTPOW	Shape number other than S1 not a power of 2
E.FILES	E.BIGFIL	Total size of file exceeds $2^{59}-1$

032

```
*** N     +          | Create a disk file (priviledged operation *)
 S
 S        •
 S
005 S1T   +          |      Input parameters:
 S
006 S0T   +          |          IP1 D: Disk accounting record number
 S
007 S0T   +          |          IP2 BD: Shape numbers S1 through Sn
 S
 S        •
 S
008 S1T   +          |     Returned parameters:
 S
009 S0T   +          |          RDAT D: Disk unique name, header block size, and disk address
 S
010 S0T   +          |               (one word)
 S
011 S0T   +          |          RCAP C: ECS file capability for new file
 S
 S        •
 S
012 S1    +          | A file of the specified dimensions (see 'Create ECS file')  is  created
 S
013       +          | in  both  ECS  and  the disk system data structure.  The resulting disk
 S
014       +          | file is opened for the creating process.  The new  file  is  associated
 S
015       +          | with the disk accounting record specified by the first parameter (IP1),
 S
016       +          | and  this  record  must  fund  all permanent disk space occupied by the
 S
017       +          | file.  Although space is reserved on the disk for the file header block
 S
018       +          | (see below), it is not copied to the disk until the file is closed.
 S
 S
019 S1    +          | For a disk file, the first level of pointers (or the data  block  in a
 S
020       +          | one  level  file)  is  associated with the header block of the file (in
 S
021       +          | contrast to the scheme for ECS files in which the file  descriptor  and
 S
022       +          | file  root  are  separate).  Thus, if the new file is a one level file,
 S
023       +          | the data block is funded and added to the ECS incarnation of the  file.
 S
 S                    ------------------
*** P     +          | * Priviledged  operations  are used by priviledged system subprocesses
*** Z     +          | and are not allowed to be in C-lists.
```

```
001 S1    +          | An ECS file capability (with write access) together with the disk
 S
002       +          | unique name and header block address are returned to the caller via the
 S
003#      +          | return parameter mechanisms (see     ). The ECS file capability has
 S
004       +          | all options set except for: OB.DSTRY, OB.CHNAM, OB.CREBL, OB.DELBL,
 S
005       +          | OB.PLMAP, OB.FDAE, and OB.TRDB. There are many restrictions on the
 S
006       +          | shape of a disk file. Considerations of efficiency and utility have
 S
007       +          | forced what may seem to be arbitrary limitations on the range of the
 S
008       +          | dimensions of a disk file.
 S
 S
009 S1    +          | Shape restrictions: one level files - data block size < 512 - 4
 S
 S
010 S1T   +          |                     multi-level files - number of levels ≤ 11
 S
012 S0U   +          |                                         data block size = 128 or 256 or
 S
013#S0T   +          |                                                           512
 S
014 S0    +          |                                         pointer block fan out ≥ 8
 S
015 S0T   +          |                                                ≥ 8
 S
016 S0T   +          |                                                ≤ 128
 S
017 S0T   +          |                                                = 2**n (for some n)
 S
018 S0    +          |                                         first shape number (S1) ≤ 128
 S
019 S0    +          |                                         total length of file ≤ 2**36 - 1
 S
 S
021 S1    +          | In addition to errors which may be detected while processing the file
 S
022       +          | description, errors may be encountered in funding the newly created
 S
023       +          | file. Fixed ECS space is funded from the process allocation block; a
 S
024       +          | one-level file receives the data block size plus 4 words; a multi-level
 S
025       +          | file receives the sum of the number of levels plus the first shape
 S
026       +          | number plus four. Permanent disk space is funded from the disk
```

accounting record (IP1); a one-level file receives from 1 to 7 sectors depending on the data block size, and a multi-level file receives one sector. Enough Disk system Data Storage (DDS) is reserved to permit creating at least one block on the file. Additional DDS space may be reserved by making the appropriate call upon the disk system.

Possible errors while creating a disk file:

Class	Number	Description
E.PARMS	E.NEGPAR	Level number < 1
E.FILES	E.LLEV	Level number too large
E.FILES	E.NEGSIZ	Data block size negative or ~~(multi-level file)~~ too small (< 128)
		Shape number too small (< 8)
E.FILES	E.BIGSIZ	Data block too big (multi-level > 512) (one-level > 508); shape number too big (> 128)
E.FILES	E.NOTPOW	Data block size or pointer block size (multi-level file) not power of 2
E.FILES	E.BIGFIL	File too big (> 2^{36} - 1)
E.FILES	E.NOLFH	Too many locally open files
E.FILES	E.FULL	Disk system tables full
E.ABLOCK	E.NOABLK	Disk accounting record does not exist
E.ABLOCK	E.NOECS	Not enough fixed ECS for file
E.ABLOCK	E.NODSK	Not enough permanent disk space in disk accounting record
E.ABLOCK	E.NODDS	Not enough DDS space

(EC:CBLK)

```
001 N     +   | Create an ECS File Data Block
S
S         •
S
002 S1T   +   |     IP1 C: Capability for file (OB.CREBL)
S
003 S0T   +   |     IP2 D: Address of block in the file
S
S
004 S1    +   | Once an ECS file has been created, data blocks of the declared length
S
005       +   | (Sn) may be added subsequently, one at a time, to hold data or code.  A
S
006       +   | count  of  the subprocess map entries which reference the data block is
S
007       +   | maintained with  each  data  block.  (This  count  is  important  when
S
008       +   | deleting  a block - see below).  To create a block, the user supplies a
S
009       +   | capability for the file to which the  block  is  being  added,  and  an
S
010       +   | address  which  is  contained  in the block which is to be added to the
S
011       +   | file.
S
S
012 S1    +   | When a data block is added to a file, it may also be necessary for  the
S
013       +   | system  to  create  some or all of the pointer blocks between that data
S
014       +   | block and the file descriptor.  Recall that pointer blocks are required
S
015       +   | to link the file descriptor to the data blocks in any  file  with  more
S
016       +   | than  one  shape  number  (i.e.,  not  a  one  level  file).  All newly
S
017       +   | allocated ECS space is charged to the allocation block associated  with
S
018       +   | the file.
```

Possible errors while creating a block:

Class	Number	Description
E.PARMS	E.NEGIX	C-list index is negative
E.PARMS	E.BIGIX	C-list index exceeds full C-list
E.PARMS	E.NEGPT	Address of new block is negative
E.PARMS	E.BIGPT	Address of new block ≥ file length (address range: 0 to length-1)
E.OPER	E.CAPTY	Type or options bad
E.FILES	E.NOFIL	File does not exist
E.FILES	E.ISBLK	Address of new block ~~indicates~~ corresponds to an already existing block
E.ABLOCK	E.NOECS	No ECS space available

```
001 N    +   | Create disk file data blocks                    (DF:CBCK)
 S
 S        •
 S
002 S1T  +   |     IP1 C: ECS file capability for locally open disk file (OB.DCRBL)
 S
003 S0T  +   |     IP2 D: Starting file address
 S
004 S0T  +   |     IP3 D: Word count
 S
 S
005 S1   +   | Data blocks may be added to any locally open disk file.  The file to be
 S
006      +   | enlarged  is  specified  by  giving the ECS file capability (IP1) for a
 S
007      +   | locally open disk file (i.e., a disk file which has been opened by  the
 S
008      +   | calling  process).  More  than one block may be added at once.  Blocks
 S
009      +   | beginning with the one  containing  the  starting  file  address  (IP2)
 S
010      +   | through  the  one  containing  the starting address plus the word count
 S
011      +   | (IP2 + IP3) are created and attached to the calling process.
 S
 S
012 S1   +   | All new data blocks and any necessary new pointer blocks are funded  on
 S
013      +   | the  disk  by the disk accounting record associated with the disk file.
 S
014      +   | Disk space for all data blocks is funded before any blocks are  created
 S
015      +   | (unused  space is returned in case of an error).  Swapped ECS space for
 S
016      +   | attaching the new data blocks is charged to the swapped ECS account  of
 S
017      +   | the  calling  process.  The  actual ECS file space occupied by the new
 S
018      +   | file blocks is absorbed by the disk system ECS allocation block.
 S
 S
019 S1   +   | The create disk file data block operation will generate an F-return  if
 S
020      +   | the  disk file to be enlarged is frozen, (i.e., all usage is being paid
 S
021      +   | for by a single user).  On one-level disk files this action ~~does only~~
 S
022      +   | ~~address checking~~.  Errors may occur after one or more blocks have been
 S
023      +   | created.  The file address of the first block not created is  added  to
```

```
001       +     | the  error  number  to  indicate  the  state of the file when the error
 S
002       +     | occurred.
 S
 S
003 S1    +     | Possible errors while creating a disk file data block:
 S
 S
005 S1T T*+     |   Class     Number       Description
 S
 S
006 S1T T*+     |   E.PARMS   NumberAR     Description
 S
 S
007 S1T T*+     |   E.PARMS   E.NEGIX      C-list index is negative
 S
008 S0T T*+     |   E.PARMS   E.BIGIX      C-list index exceeds full C-list
 S
009 S0T T*+     |   E.PARMS   E.NEGPAR     Negative starting file address or negative
 S
010 S0T   +     |                           word count
 S
011 S0T T*+     |   E.PARMS   E.BIGPAR     File address plus word count exceeds file
 S
012 S0T   +     |                           length
 S
013 S0T T*+     |   E.OPER    E.CAPTY      Type or options bad
 S
014 S0T T*+     |   E.FILES   E.NODFIL     No such disk file opened by this process
 S
015 S0T T*+     |   E.FILES   E.DIOERR[1]  Error on pointer block read from disk
 S
016 S0T   +     |                           (modifier = file address of first block not
 S
*** S0T   +     |                           created)
 S
020 S0T T*+     |   E.FILES   E.ISBLK[1]   Data block address already exists (modifier =
 S
021 S0T   +     |                           file address of block which exists)
 S
022 S0T T*+     |   E.FILES   E.NABR[1]    No attached block record, i.e., disk system
 S
023 S0T   +     |                           out of local storage (modifier = file address
 S
024 S0T   +     |                           of first block not created)
 S
025 S0T T*+     |   E.ABLOCK  E.NODSK      Insufficient disk space to fund all data blocks
 S
 S
*** F Z   +     | [1] Error may occur after zero, one or more blocks have been created.
```

032

102

001 SOT T*+ S 002 SOT +	I	E.ABLOCK E.NOSWP	Insufficient swapped ECS in process disk accounting record.

```
001 N      +        | Delete a data block from an ECS file                    (EC:DBLK)
S
S          *
S
002 S1T    +        |     IP1 C: Capability for file (OB.DELBL)
S
003 S0T    +        |     IP2 D: Address of block to be deleted
S
S
004 S1     +        | A block can be deleted from a file as long as it is not  referenced  by
S
005        +        | an  entry  in some subprocess map (reference count = 0).  The user must
S
006        +        | supply the capability index for the file and  the  address  within  the
S
007        +        | file  of  the block which is to be deleted.  If the block is referenced
S
008        +        | by a map entry, an error is issued.
S
S
009 S1     +        | Possible errors while deleting a data block from an ECS file:
S
011 S0T T*+         |   Class        Number       Description
S
S
012 S1T T*+         |   E.PARMS      E.NEGIX      C-list is negative
S
013 S0T T*+         |   E.PARMS      E.BIGIX      C-list index exceeds full C-list
S
014 S0T T*+         |   E.PARMS      E.NEGPAR     Pointer is negative
S
015 S0T T*+         |   E.PARMS      E.BIGPAR     Pointer is too large
S
016 S0T T*+         |   E.OPER       E.CAPTY      Type or options bad
S
017 S0T T*+         |   E.FILES      E.NOBLK      Block to be deleted does not exist
S
018 S0T T*+         |   E.FILES      E.INMAPS     Block to be deleted is in a map
```

```
001 N      +    | Delete a block from a disk file
 S
 S
002 S1T    +    |     IP1 C: ECS file capability for locally open disk file (OB.DLBL)
 S
003 S0T    +    |     IP2 D: Starting address in file
 S
004 S0T    +    |     IP3 D: Word count
 S
 S
005 S1     +    | Data blocks may be deleted from a locally open disk file.  The file is
 S
006        +    | specified  by  giving  the ECS file capability (IP1) for a locally open
 S
007        +    | disk file.  More than one block may be deleted at a time.  Blocks  are
 S
008        +    | destroyed beginning with the block containing the starting file address
 S
009        +    | (IP2)  through  the block containing the starting address plus the word
 S
010        +    | count (IP2 + IP3).
 S
 S
011 S1     +    | The blocks to be deleted may be either on the disk (unattached)  or  in
 S
012        +    | ECS.   If  in  ECS,  they must be attached only to the calling process.
 S
013        +    | Furthermore, they may not be in any maps.
 S
 S
014 S1     +    | The disk space occupied by the deleted data blocks is refunded  to  the
 S
015        +    | disk accounting record associated with the file.  Space for any pointer
 S
016        +    | blocks  which  may  become  empty  is  not  refunded  until the file is
 S
017        +    | globally closed (no longer in use) or pseudo-closed  (a  technique  for
 S
018        +    | updating  the disk version of a file).  If some of the data blocks were
 S
019        +    | locally attached, swapped ECS space is refunded to the calling program.
 S
 S
020 S1     +    | The operation to delete disk file data blocks will F-return if the disk
 S
021        +    | file is frozen.  This action does only address checking  on  one-level
 S
022        +    | disk  files.  An  error  may  occur after one or more blocks have been
 S
023        +    | deleted.  The file address of the first block not deleted is  added  to
```

001 + | the error number to facilitate error recovery.
S
S •
S
002 S1 + | Possible errors while deleting a block from a disk file:
S
S
004 S1AT *+ | Class Number Description
S
S
005 S1T T*+ | E.PARMS E.NEGIX C-list index is negative
S
006 SOT T*+ | E.PARMS E.BIGIX C-list index exceeds full C-list
S
007 AT T U+ | E.PARMS E.NEGPAR Negative starting file address or nega-
S
008 + | tive word count
S
009 AT T U+ | E.PARMS E.BIGPAR File address plus word count exceeds file
S
010 + | length
S
011 AT T U+ | E.OPER E.CAPTY Type or options bad
S
012 AT T U+ | E.FILES E.NODFIL No such disk file open by calling process
S
013 AT T U+ | E.FILES E.DIOERR² I/O error reading pointer block (modifier
S
014 + | = file address of first block not
S
015 + | deleted)
S
*** + |
S
020 AT T U+ | E.FILES E.NOBLK² Block to be deleted does not exist (modi-
S
021 + | fier = file address of first block which
S
022 + | does not exist)
S
023 AT T U+ | E.FILES E.TMGA² Block is attached to another process
S
024 + | (modifier = file address of attached
S
025 + | block)
S
026 AT T U+ | E.FILES E.INMAPS² Block in some map in calling process
S
S ----------
*** F Z + | ² Error may occur after zero, one or more block have been deleted.

001 + | (modifier = file address of block in map)

E. FILES E. FLEV attempt to create data block of a non-linked file

001 N +
S
S •
S
002 S1T +
S
S
003 S1 +
S
004 +
S
005 +
S
S
006 S1 +
S
S
007 S1T T*+
S
S
008 S1T T*+
S
009 S0T T*+
S
010 S0T T*+
S
011 S0T T*+

Moore Business Forms, Inc. sv

Delete a Disk file (OF: DFIL)

IPI capability for a disk file (OB: DSTRF)

The disk file to be deleted must be locally open and it must not be opened by any other process. Blocks may exist in the file and may be attached but not. Block of the file may be in a imap. If anything is OK, mapped and file buffers are immediately refunded to the local accounting record. Disk space occupied by the file will be reclaimed by a parallel process which will complete the file deletion. and the ECS incarnation of the file

Possible reasons for deleting a disk file

E. open	E. captive	type of process back
E. files	E. modified	no such locally open disk file
E. files	E. TIMSON	file to open in some other process
E. files	E. in maps	block of file is in a map (modifier = file address of offending block)

```
001 N      +          | Move an ECS File Block                                    (EC.MBCK)
 S
 S         •
 S
002 S1T    +          |     IP1 C: Capability for source file (OB.RDFIL, OB.DELBL)
 S
003 S0T    +          |     IP2 D: Address in source file of block to be moved
 S
004 S0T    +          |     IP3 C: Capability for destination file (OB.WFILE, OB.CREBL)
 S
005 S0T    +          |     IP4 D: Address in destination file of block to be moved
 S
 S
006 S1     +          | File blocks can be transferred between ECS files whose data block sizes
 S
007        +          | (Sn) are equal. In addition to the capabilities for the source and
 S
008        +          | destination files, the system expects to receive from the user the
 S
009        +          | address within the source file of the block to be moved and the address
 S
010        +          | in the destination file to which the block is being moved. Any address
 S
011        +          | within each block suffices. If the block to be moved is referenced by
 S
012        +          | a map, moving it (which deletes it from the source file) would cause
 S
013        +          | problems when swapping; therefore an error is issued. If no errors
 S
014        +          | occur, the designated block is deleted from the source file and added
 S
015        +          | to the destination file. The contents of the block are not affected.
 S
 S
016 S1     +          | Possible errors while moving a block:
 S
 S
018 S1AT *+           |     E.FILES   E.NOBLK        Block to be moved does not exist in
 S
019        +          |                              source file
 S
020 AT T U+           |     E.FILES   E.ISBLK        A block already exists at the designated
 S
021        +          |                              address in the destination file
 S
022 AT T U+           |     E.FILES   E.MISMCH       Files do not have equal data block sizes
 S
023 AT T U+           |     E.FILES   E.INMAPS       Block to be moved is in a map
 S
024 AT T U+           |     E.PARMS   E.NEGPT        File address negative
```

001 AT T U+ S		E.PARMS	E.BIGPT	File address too large
002 AT T U+ S		E.PARMS	E.NEGIX	C-list index is negative
003 AT T U+ S		E.PARMS	E.BIGIX	C-list index exceeds full C-list
004 AT T U+		E.OPER	E.CAPTY	Type or options bad

-111-

(DF:OPRO, DF:OPRW)

001 N + | Open a disk file read-only (read/write)
S
S
002 S1T + | Input parameter:
S
003 S0T + | IP1 C: Disk file capability (OB.OPEN (OB.WFILE))
S
S
004 S1T + | Returned parameter:
S
005 S0T + | RCAP C: ECS file capability for open disk file
S
S
006 S1 + | Before a disk file can be manipulated, it must first be logically
S
007 + | opened. The open action returns an ECS file capability for the ECS
S
008 + | incarnation of the disk file. Almost all disk file operations require
S
009 + | this ECS file capability as a parameter. To facilitate file sharing, a
S
010 + | disk file may be opened by several processes and may be opened
S
011 + | repeatedly by a single process. A process local open count permits
S
012 + | repeated opening and closing by a single process without interferring
S
013 + | with the global open count, which reflects the number of separate
S
014 + | processes which have logically opened the file.
S
S
015 S1 + | The disk file capability (IP1) contains the disk address and unique
S
016 + | identification for the disk file. The first time a disk file is
S
017 + | opened, the header block for the file must be read from the disk and an
S
018 + | ECS file created from the shape description contained therein. Subse-
S
019 + | quent opens return a capability for the ECS file created by the initial
S
020 + | open without requiring a disk reference.
S
S
021 S1 + | If the file has not been opened previously by the calling process, the
S
022 + | process is charged for the fixed ECS space required for the ECS file
S
023 + | descriptor and the first level of the file tree. On the disk, the

If the file has previously been "opened" by another process, the open action will not create a new file descriptor.

```
001       +   | first level of the file is stored in the header block of the  file  and
S
002       +   | must  be brought to ECS when the file is opened.  If the file is frozen
S
003       +   | or is already open in the calling process, no charge is made for  fixed
S
004       +   | ECS  space.  Fixed ECS space is funded by the process allocation block;
S
005       +   | a one-level file must pay for space equal to the data block  size  plus
S
006       +   | four;  a  multi-level file pays for space equal to the number of levels
S
007       +   | plus the number of branches from the first level (S1) plus four.
S
S         •   (disk system global storage)
S
008 S1    +   | An ECS file capability, with  or  without  write  access,  is  returned
S
009       +   | depending  on  whether  the open read only or open read/write operation
S
010       +   | was called for.  The "open read/write" operation requires  an  additional
S
011       +   | option  bit (OB.WFILE) in the disk file capability (IP1) to satisfy the
S
012       +   | ECS system parameter checking.  In addition, the option bits of the ECS
S
013       +   | file capability are ANDed with the option bits  of  the  user  supplied
S
014       +   | disk  file  capability (IP1) before the ECS file capability is returned
S
015       +   | to the caller.  Finally the following options are turned off in the  ECS
S
016       +   | file  capability  which  is  returned:  OB.DSTRY,  OB.CHNAM, OB.CREBL,
S
017       +   | OB.DELBL, OB.PLMAP, OB.FDAE, and OB.TRDB.
```

In addition, sufficient DDS space is reserved to allow the attaching of any one file block. additional DDS space may be reserved explicitly or automatically as needed.

"Open read/write" returns extra options OB.WFILE, OB.DCRBL, and OB.ODABL.

032 102

001 N + | Prohibited options are:
S
S •
S
003 S1AT U+ | OB.DSTRY Destroy a file
S
004 S0AT U+ | OB.CHNAM Change unique name
S
005 S0AT U+ | OB.CREBL Create a block
S
006 S0AT U+ | OB.DELBL Delete a block
S
007 S0AT U+ | OB.FDAE Direct ECS access
S
008 AT U + | OB.PLMAP Place in subprocess map
S
009 AT U + | OB.TRDB Test and reset dirty bit
S
S F-return condition: file is "exclusively" open already
011 S1 + | Possible errors while opening a disk file:
S
013 S0AT *+ | Class Number Description
S
S
014 S1AT *+ | E.PARMS E.NEGIX C-list is negative
S
015 S0AT *+ | E.PARMS E.BIGIX C-list index exceeds full C-list
S
016 AT T U+ | E.FILES E.NODFIL No such disk file or
S
017 S0 + | Disk parity error on header block
S
018 S0AT *+ | E.FILES E.TMOPN Too many opens (local open count >
S
019 + | $2^{18}-1$ or global open count > $2^{18}-1$)
S
020 S0AT *+ | E.FILES E.NOLFH Too many locally open files (> 25)
S
021 AT T U+ | E.FILES E.FULL Disk system tables full
S
022 AT T U+ | E.OPER E.CAPTY Type or options bad
S
023 S0AT *+ | E.ABLOCK E.NOECS Insufficient fixed ECS to open file
E.ablock E.NOOPS Insufficient PPS space

(DF:CLO - ECS file capability)
(DF:CLOS - Disk file capability)

```
001 N      +        | Close a disk file
 S
 S         •
 S
002 S1T    +        |      IP1 C: ECS file capability (or disk file capability)
 S
003 S0T    +        |           for locally open disk file (OB.CLOSE)
 S
 S
004 S1     +        | A disk file should be closed when a process is through manipulating it.
 S
005        +        | If  the  calling process is the only process to have opened the file, a
 S
006        +        | close will cause the contents of the file on the disk to be updated  to
 S
007        +        | reflect  all  changes  in the file content and size.  Should the system
 S
008        +        | crash after  a  successfully  completed  close,  it  is  unlikely  that
 S
009        +        | information  in  or about the file will be lost.  On the other hand, as
 S
010        +        | long as the file remains open by some process, all changes  made  since
 S
011        +        | the file was first opened may be lost in the event of a crash.
 S
 S
012 S1     +        | As  mentioned  above, a process local open count records multiple opens
 S
013        +        | by a single process.  If a close decrements the open count to zero, any
 S
014        +        | blocks of the file which are attached to the process are detached,  and
 S
015        +        | the  swapped  ECS space is refunded to the process.  Fixed ECS space is
 S
016        +        | also refunded, unless the file was frozen when the process opened it.
 S
 S
017 S1     +        | If the local open count becomes zero, any local claims on the file  are
 S
018        +        | released  and  the  global  open count is decremented.  When the global
 S
019        +        | open count becomes zero, a request is sent to  a  special  disk  system
 S
020        +        | process  which  will  update the contents of the file on the disk.  The
 S
021        +        | update is carried on simultaneously with the execution of  the  process
 S
022        +        | requesting the close file action.  The updating of the file on the disk
 S
023        +        | consists  of  re-writing  any  pointer  blocks  which point directly or
```

```
001       +   | indirectly to data blocks which have been modified and  written  to  new
S
002       +   | locations on the disk.  After all pointer blocks have been written, the
S
003       +   | root  level  pointers  are re-written with the file header at their old
S
004       +   | address on the disk.  In this way, the old contents of  the  file  will
S
005       +   | not  be  lost if the system should crash before the update procedure is
S
006       +   | complete.
S
S         •
S
007 S1    +   | Possible errors while closing a disk file:
S
S
008 S1T T*+   |     Class       Number          Description
S
S
009 S1T T*+   |     E.PARMS     E.NEGIX         C-list index is negative
S
010 S0T T*+   |     E.PARMS     E.BIGIX         C-list index exceeds full C-list
S
011 S0T T*+   |     E.OPER      E.CAPTY         Type or options bad
S
012 S0T T*+   |     E.FILES     E.NODFIL        No such locally open file
S
013 S0T T*+   |     E.FILES     E.INMAPS        Attached block in a map (modifier =
S
014 S0T   +   |                                  file address of block in map)
```

```
001 N       +   | Attach file block(s)                                  (DF:ATCH)
 S
 S          •
 S
002 S1T     +   |     IP1 C: ECS file capability for a locally open disk file (?)
 S
003 S0T     +   |     IP2 D: Starting address in file
 S
004 S0T     +   |     IP3 D: Word count
 S
 S          •
 S
005 S1      +   | Data blocks of a disk file may be attached by a process.  Attaching a
 S
006         +   | data  block  will  cause the block to be held in ECS until the block is
 S
007         +   | detached.  If the data block is not already attached, a  disk  read  is
 S
008         +   | initiated  to bring the block from the disk.  However, the process does
 S
009         +   | not wait  for  the  block  to  arrive  from  the  disk.  Since  several
 S
010         +   | processes  may  share  a  file  and  attach  the  same  block, a global
 S
011         +   | attachment count is maintained for disk file data blocks.  If the block
 S
012         +   | being attached is already in ECS by virtue of being  attached  by  some
 S
013         +   | other  process,  no  disk  read  need be initiated.  To permit multiple
 S
014         +   | attaches by a single process, a  local  attachment  count  is  used  to
 S
015         +   | prevent one user from over-riding the attaches of another user.  Unless
 S
016         +   | the  file  (IP1) is "frozen", the user is charged for swapped ECS space
 S
017         +   | for each block which was not previously locally attached.
 S
 S
018 S1      +   | Several data blocks may be attached by one call  to  the  disk  system.
 S
019         +   | All  data  blocks  from  the block containing the starting file address
 S
020         +   | (IP2) through the data block containing the starting file address  plus
 S
021         +   | the  word  count  (IP2 + IP3) are attached.  If IP1 is a one level file,
 S
022         +   | only parameter checking is performed since the first level  of  a  disk
 S                                                  (i.e. attached)
023         +   | file is brought to ECS when the file is opened.
```

032

Errors may occur in the process of attaching a sequence of data blocks. If the problem is a non-existant block, an F-return is made to the caller. The address of the missing data block is returned in X6. In this case, the file address of the block causing the error will be added to the error number. All blocks preceding the error block will already be attached.

Possible errors on attaching file block(s):

Class	Number	Description
E.PARMS	E.NIGIX	C-list index is negative
E.PARMS	E.BIGIX	C- list index exceeds full C-list
E.PARMS	E.NEGPAR	Negative starting file address or negative word count
E.PARMS	E.BIGPAR	Starting file address plus word count too big
E.OPER	E.CAPTY	Type or options bad
E.FILES	E.NABR[3]	No attached block records (disk system out of local storage) (modifier = file address of last block attached)
E.FILES	E.TMA[3]	Too many local attaches (> 2^{12} - 1) (modifier = file address of last block attached)
E.FILES	E.TMGA[3]	Too many global attaches (> 2^8 - 1)
E.FILES	E.DIOERR[3]	Disk I/O error (modifier = file address of last block attached)

[3] May occur after zero, one, or more blocks are attached.

032

102

001 AT T U+ S	|	E.ABLOCK E.NOSWP	Insufficient swapped ECS in process disk
002 +	|		accounting record
		E.ABLOCK E.NCDDS	out of DDS space

```
001 N    +    | Detach Disk File Blocks
 S
 S       •
 S
002 S1T  +    |     IP1 C: ECS file capability for a locally
 S
003#S0T  +    |          open disk file (OB.    )
 S
004 S0T  +    |     IP2 D: Starting file address
 S
005 S0T  +    |     IP3 D: Word count
 S
 S
006 S1   +    | When portions of a disk file are no longer needed in ECS, the
 S
007      +    | corresponding blocks of the file should be detached. In detaching a
 S
008      +    | disk file block, the local attachment count is decremented. If the
 S
009      +    | local attachment count goes to zero, the global attachment count for
 S
010      +    | the block is decremented. If the global count becomes zero, the block
 S
011      +    | is removed from the ECS incarnation of the disk file. If the block is
 S
012      +    | clean (i.e., the dirty bit on the ECS file block is not set), the block
 S
013      +    | does not need to be written to the disk. Otherwise, the block is
 S
014      +    | written to the first available position on the disk. The newly written
 S
015      +    | block will not be linked (on the disk) to its file header until a close
 S
016      +    | or pseudo-close is performed on the file.
 S
 S
017 S1   +    | Unless the file is frozen, swapped ECS space will be refunded on all
 S
018      +    | blocks for which the local attachment count has become zero. As with
 S
019      +    | "attach", more than one block may be detached by a single call on teh
 S
020      +    | disk subsystem. Also, errors may occur after one or more blocks have
 S
021      +    | been detached. In this case, the file address of the offending block
 S
022      +    | is added to the error number.
 S
 S
023 S1   +    | Possible errors while detaching a file:
```

	Class	Number	Description
001 AT T U+ S			
002 AT T U+ S	E.PARMS	E.NEGIX	C-list index is negative
003 AT T U+ S	E.PARMS	E.BIGIX	C-list index exceeds full C-list
004 AT T U+ S	E.PARMS	E.NEGPAR	Negative starting file address or
005 SO + S			Negative word count
006 AT T U+ S	E.PARMS	E.BIGPAR	Starting file address plus word count too
007 + S			big
008 AT T U+ S	E.OPER	E.CAPTY	Type or options bad
009 AT T U+ S	E.FILES	E.NATH	Block not locally attached (modifier =
010 + S			file address of block)
011 AT T U+ S	E.FILES	E.DIOERR	Disk I/O error on block (modifier = file
012 + S			address of block)
013 AT T U+ S	E.FILES	E.TMD	Too many detaches (block is in a map and
014 + S			non-map attach count already zero) (modi-
015 +			fier = file address of block)

001 N + | Interchange file contents ("shazam")
S
S
002 S1I5J*+ | IP1 C: ECS file capability for primary file (OB.DDRBL, OB.PDLBL)
S
003 S0I5J*+ | IP2 C: ECS file capability for secondary file (OB.DDRBL,
S
004 + | OB.DDLBL)
S
S
005 S1 + | It is possible to interchange the contents of two disk files without
S
006 + | copying data from one file to the other. Whenever one file is
S
007 + | considered to be the "backup" version of another file containing the
S
008 + | updated version of the file, the file interchange action may be used to
S
009 + | securely update the backup file. The file interchange is performed by
S
010 + | first updating the contents of both files on the disk and then
S
011 + | interchanging the root pointers of the two files in ECS and on the
S
012 + | disk. The order of disk writes is such that first the secondary file
S
013 + | header is clobbered. Then the primary file is updated to contain the
S
014 + | form contents of the secondary file. Finally, the secondary file
S
015 + | header is updated on the disk to point to what was formerly the
S
016 + | contents of the primary file. By using this algorithm, a system
S
017 + | failure at any point will leave the primary file either unchanged or
S
018 + | containing the contents of the secondary file. The secondary file will
S
019 + | be 1) unchanged, 2) gone, or 3) will contain the previous contents of
S
020 + | the primary file. After the interchange, all blocks of both files will
S
021 + | be "detached" (i.e., on the disk).
S
S
022 S1 + | In order to interchange two files, several conditions must be met.
S
023 + | Both files must be locally open and must not be opened by any other
S
024 + | process. No block of either file may be in a map and the accounting

```
001       +    | records of the smaller file must fund the increase in file size.  The
 S
002       +    | accounting record of the larger file will be refunded an appropriate
 S
003       +    | amount of space.  One level files cannot participate in a file
 S
004       +    | interchange.
 S
 S        •
 S
005 S1 T *+    | ~~Posc~~Class~~error~~Numbere ~~in~~Description ~~file contents~~
 S
007 S1T T*+    |     E.OPER    E.CAPTY    Type or options bad
 S
008 AT T U+    |     E.PARMS   E.NEGIX    C-list index is negative
 S
009 AT T U+    |     E.PARMS   E.BIGIX    C-list index exceeds full C-list
 S
010 AT T U+    |     E.FILES   E.NODFIL   File not locally open disk file (modifier=1 if
 S
011       +    |                          secondary file; 0 if primary file)
 S
012 AT T U+    |     E.FILES   E.INMAPS   Block in a map (modifier=file address of
 S
013       +    |                          offending block)
 S
014 AT T U+    |     E.FILES   E.TMOPN    File is open in another process (modifier=1 if
 S
015       +    |                          secondary file; 0 if primary file)
 S
016 AT T U+    |     E.FILES   E.ZLEV     File is one level file (modifier=1 if secon-
 S
017       +    |                          dary file; 0 if primary file)
 S
018 AT T U+    |     E.ALLOC   E.NODSK    Insufficient disk space in file accounting
 S
019       +    |                          record
```

Possible errors while interchanging file contents:

```
001 N     +                    Read (write) file
S
S         •
S
002 S1T   +     I  IP1 C: Capability for file (OB.RDFIL,(OB.WFILE))
S
003 S0T   +     I  IP2 D: Address in file
S
004 S0T   +     I  IP3 D: Address in Central Memory
S
005 S0T   +     I  IP4 D: Count of words to be transferred
S
S
006 S1    +     I     The action of reading (writing) an ECS file
S
007       +     I     transfers words between the address space of
S
008       +     I     the running (current) subprocess and the data
S
009       +     I     blocks of a file.  In addition to the capabi-
S
010       +     I     lity index for the file, the user specifies
S
011       +     I     the address in the file of (for) the desired
S
012       +     I     information, the address in Central Memory of
S
013       +     I     the area to be read into (written from), and
S
014       +     I     the number of words that are to be read
S
015       +     I     (written).  If a transfer is requested which
S
016       +     I     involves a file address corresponding to a
S
017       +     I     non-present data block, the transfer proceeds
S
018       +     I     until the non-present file address is encoun-
S
019       +     I     tered, whereupon F-return action passes con-
S
020       +     I     trol to the disk system, which supplies missing
S
021       +     I     blocks and reinitiates the actcontrol to the
S
022       +     I     disk system.  The disk system will check to
S
023       +     I     see if the file (IP1) is a locally open disk
S
024       +     I     file.  If not, an F-return will again be
```

```
001       +                 |                    initiated (see "Process control" and "Opera-
S
002       +                 |                    tions").  If the file is a disk file, the
S
003       +                 |                    missing blocks will be fetched from the disk,
S
004       +                 |                    the transfer performed, and the blocks
S
005       +                 |                    released or returned to the disk.  If a block
S
006       +                 |                    which does not exist in the disk version of
S
007       +                 |                    the file is encountered, an F-return is
S
008       +                 |                    initiated.
S
S         •
S
009 S1    +                 |                    Possible errors while reading (writing) a
S
010       +                 |                    file:
S
S
012 S1AT *+                 |  Class     Number          Description
S
S
013 S1AT *+                 |  E.PARMS   E.NEGIX         C-list index is negative
S
014 AT T U+                 |  E.PARMS   E.BIGIX         C-list index exceeds full C-list
S
015 AT T U+                 |  E.PARMS   E.NEGPAR        Word count negative
S
016 S0AT *+                 |  E.PARMS   E.NEGPT         File address negative or
S
017 S0    +                 |                            CM address negative
S
018 S0AT *+                 |  E.PARMS   E.BIGPAR        File address plus word count (IP2 + IP4)
S
019       +                 |                            exceeds file length or
S
020 S0    +                 |                            CM address plus word count (IP3+IP4)
S
021       +                 |                            exceeds user's field length
S
022 AT T U+                 |  E.OPER    E.CAPTY         Type or options bad
```

Make an exclusive claim on a file (with or without wait) (DF:ECLM) (with wait) (DF:ECLF) (without wait)

IP1 C: ECS file capability for a locally open disk file (OB.ECLM)

An exclusive claim may be honored only when no other claim (exclusive or shared) has been issued for the file in question. ~~Otherwise~~ the process is added to the end of the claim wait queue for the file (IP1) if the claim cannot be satisfied immediately.

This queue is used to keep track of claims which cannot be honored due to outstanding claims on the file. Whenever a claim is released (see below), the claim wait queue is checked. If non-empty, either a single exclusive claim is honored from the head of the queue provided no other claims exist on the file, or a sequence of shared claims is processed from the beginning of the queue. To avoid locking up a user process in the disk system, a time limit is enforced in the claim wait queue. If a claim cannot be satisfied after a wait of one to two minutes in the queue, the user is removed from the queue and an error is returned to the caller. Finally, the claim mechanisms permit only one claim per user. Any attempt by a single user to make more than one claim on a single file will cause an error to be returned.

Possible errors while making an exclusive claim:

Class	Number	Description
E.PARMS	E.NEGIX	C-list index negative
E.PARMS	E.BIGIX	C-list index too large
E.OPER	E.CAPTY	Type or options bad
E.FILES	E.NOFIL	No such open disk file

032

001 S0AT *+ S		E.FILES	E.EXCLAM	Local exclusive claim already
002 S0AT *+ S		E.FILES	E.SHCLAM	Local shared claim already
003 S0AT *+		E.FILES	E.TIMOUT	Time out on claim wait queue

001 N + | Make a shared claim on a file (with or without wait) CDF:SCLM)
S (DF:SCLF)
S •
S
002 S1I5J*+ | IP1 C: ECS file capability for a locally open disk (OB.SCLM)
S
S
003 S1 + | If the file is not exclusively claimed by some other process and no
S
004 + | process is waiting for an exclusive claim, a shared claim is honored.
S
005 + | Note that the shared claim can be honored for several processes and has
S
006 + | the effect of preventing an exclusive claim. If the claim cannot be
S
007 + | honored, the user is added to the end of the claim wait queue, or passed an immediate [illegible] depending on the type of [illegible] call (wait or no wait). (See
S
008 + | above.)
S
S
009 S1 + | Possible errors while make a shared claim:
S
S
011 S1AT *+ | Class Number Description
S
S
012 S1AT *+ | E.PARMS E.NEGIX C-list index negative
S
013 S0AT *+ | E.PARMS E.BIGIX C-list index too large
S
014 S0AT *+ | R.OPER E.CAPTY Type or options bad
S
015 S0AT *+ | E.FILES E.NODFIL No such open disk file
S
016 S0AT *+ | E.FILES E.EXCLAM Local exclusive claim already
S
017 S0AT *+ | E.FILES E.SHCLAM Local shared claim already
S
018 S0AT *+ | ~~E.FILES E.TIMOUT Time out on claim wait queue~~

```
001 N      +        | Release claim on a file                                    (DF:RLSE)
 S
 S         •
 S
002 S1I5J*+         |      IP1 C:  ECS file capability for a locally open disk file (OB.REL)
 S
 S
003 S1     +        | If there is a claim (shared or exclusive) on the file by the calling
 S
004        +        | process, the file's claim status is updated to reflect the release of
 S
005        +        | the claim.  If the claim was an exclusive claim or the last shared
 S
006        +        | claim on the file, the claim wait queue is processed.  Either one
 S
007        +        | exclusive claimer or a sequence of shared claimers (whichever occurs
 S
008        +        | first) are activated from the claim wait queue.
 S
 S
009 S1     +        | Possible errors while releasing a claim:
 S
 S
011 S1AT *+         |      Class      Number          Description
 S
 S
012 S1AT *+         |      E.PARMS    E.NEGIX         C-list index negative
 S
013 S0AT *+         |      E.PARMS    E.BIGIX         C-list index too large
 S
014 S0AT *+         |      E.OPER     E.CAPTY         Type or options bad
 S
015 S0AT *+         |      E.FILES    E.NODFIL        No such open disk file
 S
016 S0AT *+         |      E.FILES    E.NOCLAM        No local claim exists
```

```
001 N      +          | Read shape of an ECS file                                    (EC:RS HP)
  S
  S
002 S1T    +          |     IP1 C: Capability for file
  S
003 S0T    +          |     IP2 D: Address of buffer for the shape numbers
  S
004 S0T    +          |     IP3 D: Buffer size
  S
  S
005 S1     +          | The  shape  of  a  file is described by a sequence of positive integers
  S
006        +          | (S1,S2,...,Sn), each of which is the number of  branches  in  the  file
  S
007        +          | tree  at  each  node of level i (1 ≤ i ≤ n).  Each Si (i > 1) must be a
  S
008        +          | non-negative power of two.  The user can obtain these shape numbers  by
  S
009        +          | specifying  the  index  of  the  capability for the file whose shape he
  S
010        +          | wants to read, and the address and size  of  a  buffer  for  the  shape
  S
011        +          | numbers.   The number of levels in the file is placed in the first word
  S
012        +          | of  the  buffer  and  the  shape  numbers  (S1,...,Sn)  are  placed  in
  S
013        +          | succeeding  words  until  either  the  buffer  is full or all the shape
  S
014        +          | numbers have been passed.
  S
  S
015 S1     +          | Possible errors while reading shape:
  S
  S
017 S1AT   *+         |     Class      Number         Description
  S
  S
018 S1AT   *+         |     E.FILES    E.NOFIL        File whose shape is to be read  does  not
  S
019        +          |                               exist
  S
020 AT T U+           |     E.PARMS    E.NEGIX        C-list index negative
  S
021 AT T U+           |     E.PARMS    E.BIGIX        C-list index exceeds full C-list
  S
022 AT T U+           |     E.PARMS    E.NEGPT        Buffer address is negative
  S
023 S0AT   *+         |     E.PARMS    E.NEGPAR       Buffer size ≤ 0
  S
024 S0AT   *+         |     E.PARMS    E.BIGPAR       Buffer  address  plus size exceeds user's
```

032


```
001        +                |                                  field length
 S
002 AT T U+                 |     E.OPER     E.CAPTY           Type or options bad
```

032

```
001 N      +    | Check for missing ECS data blocks ("probe")          (EC:PROB)
 S
 S         •
 S
002 S1T    +    |     IP1 C: Capability for ECS file
 S
003 S0T    +    |     IP2 D: Address of block in ECS file
 S
 S
004 S1     +    | This action allows the user to check for the presence of a data  block.
 S
005        +    | The parameters required are the index of the capability for the file to
 S
006        +    | which  the  block  belongs,  and  the address within the file where the
 S
007        +    | block is supposed to be located.  The number of missing levels  in  the
 S
008        +    | path  from  the  root  of  the  file  tree  to that particular block is
 S
009        +    | returned in register X6.  Thus, if the block is present, X6  <-  0;  if
 S
010        +    | the  n  level  file  is  empty,  X6 <- n; and if only the data block is
 S
011        +    | missing (its pointer block is present), X6 <- 1.
 S
 S
012 S1     +    | Possible errors while checking for missing blocks:
 S
 S
014 S1AT  *+    |     Class      Number        Description
 S
 S
015 S1AT  *+    |     E.FILES    E.NODFIL      The file does not exist
 S
016 AT T U+     |     E.OPER     E.CAPTY       Type or options bad
 S
017 AT T U+     |     E.PARMS    E.NEGIX       C-list index is negative
 S
018 AT T U+     |     E.PARMS    E.BIGIX       C-list index exceeds full C-list
 S
019 AT T U+     |     E.PARMS    E.NEGPAR      The address of the block is negative
 S
020 AT T U+     |     E.PARMS    E.BIGPAR      The address of the block is too large
```

032

```
001 N      +    | Close all open files                          (DF:CAOF)
 S
 S         •
 S
002 S1AT   +    |     No input parameters
 S
 S
003 S1     +    | This action closes all  locally  open  files  which  do  not  have  the
 S
004        +    | close-all-over-ride flag set.  It is to be used primarily to clean up
 S
005        +    | the user process between major job steps or after  things  have  gotten
 S
006        +    | fouled up.  The action will abort if a block is in the map of some file
 S
007        +    | which is being closed.
 S
 S
008 S1     +    | Possible errors while closing all open files:
 S
010 AT T U+     |     E.FILES    E.INMAPS        Some block of some file is in a map (file
 S
011        +    |                                address added to error number)
```

```
001 N      +          | Set/reset close all over-ride (priviledged operation)
 S
 S         •
 S
002 S1T    +          |     Input parameters:
 S
 S
003 S1I10*+           |           IP1 C:  ECS file capability for an open disk file
 S
004 S0I10*+           |           IP2 D:  Zero for reset; non-zero for set
 S
 S
005 S1T    +          |     Returned parameter:
 S
 S
006 S1I10*+           |           X6 is set to the old value of the over-ride flag
 S
 S
007 S1     +          | The setting of the flag which over-rides the close-all action is
 S
008        +          | controlled by this action. If a file should remain open through the
 S
009        +          | close-all action, the over-ride flag must be set.  If IP2 is zero, the
 S
010        +          | over-ride flag is reset.  Otherwise, the over-ride flag is set to
 S
011        +          | prevent the file from being closed by the "close-all open files"
 S
012        +          | action.
 S
 S
013 S1     +          | Possible errors while setting-resetting close-all over-ride:
 S
015 AT T U+           |     Class      Number          Description
 S
016 AT T U+           |     E.FILES    E.NODFIL        No such locally open disk file
 S
017 AT T U+           |     E.PARMS    E.NEGIX         C-list index is negative
 S
018 AT T U+           |     E.PARMS    E.BIGIX         C-list index exceeds full C-list
 S
019 AT T U+           |     E.OPER     E.CAPTY         Type or options bad
```

(DF:CFLG)

For a file to

```
001 N      +     | Check for missing disk file blocks ("probe")
 S
 S        •
 S
002 S1I5J*+      |      IP1 C:  ECS file capability for a locally open disk file
 S
003 S0I5J*+      |      IP2 D:  Address of a block in the file
 S
 S
004 S1     +     | The existence of data blocks and pointer blocks in disk  files  can  be
 S
005        +     | checked with this action.  The result is returned in register X6 and is
 S
006        +     | interpreted  the  same  as  for  the "probe" of an ECS file.  The first
 S
007        +     | level of a disk file always exists, and thus  the  empty  n-level  disk
 S
008        +     | file  would return X6 <- n-1 for all file addresses.  Disk file "probe"
 S
009        +     | may indicate fewer missing levels than there really are if blocks  have
 S
010        +     | been  deleted since the last "close" or "pseudo-close" of the file.  An
 S
011        +     | I/O error on a data or pointer block is treated as  if  the  block  were
 S
012        +     | missing from the file.
 S
 S
013 S1     +     | Possible errors during a disk file block probe:
 S
015 AT T U+      |      Class      Number_        Description
 S
016 AT T U+      |      E.PARMS    E.NEGIX        C-list index is negative
 S
017 AT T U+      |      E.PARMS    E.BIGIX        C-list index exceeds full C-list
 S
018 AT T U+      |      E.OPER     E.CAPTY        Type or options bad
 S
019 AT T U+      |      E.FILES    E.NODFIL       No such locally open file
 S
020 AT T U+      |      E.FILES    E.BIGPAR       File address too big
 S
021 AT T U+      |      E.FILES    E.NEGPAR       File address negative
```

(DF:PROB)

move up to follow ECS "probe"

```
001 N      +          | Test and reset dirty bit                                    (EC: TRDB)
 S
 S
002 S1T    +          |     IP1 C: Capability for file (OB.TRDB)
 S
003 S0T    +          |     IP2 D: Any address within block to be tested
 S
 S
004 S1I5J5+           |      Returns  0  in  X6 if the block was clean.  Returns 1 in X6 if the
 S
005        +          |      block was dirty.  F-returns if specified block does not exist.
 S
 S
006 S1     +          | A bit on each data block of a file is used to tell whether or  not  the
 S
007        +          | block  has  been  written  in  since  it  was  last tested.  A complete
 S
008        +          | description of the logic controlling the bit is
 S
 S
009 S1I5   +          |      1.  Data blocks are created clean.
 S
010 S0I5   +          |      2.  Blocks are dirtied by:
 S
011 S0I10*+           |           a.  file writes to any part of the  block,  including  writes
 S
012        +          |               with a word count of 0;
 S
013 S0I10*+           |           b.  being put in a map R/W;
 S
014 S0I10*+           |           c.  being put in a DAE map entry
 S
015 S0I5J5+           |      3.  Move block carries the dirty bit along with the block.
 S
016 S0I5J5+           |      4.  Test and reset leaves the block clean.
 S
 S
017 S1     +          | Possible errors while testing and resetting dirty bit:
 S
 S
019 S1AT *+           |      Class      Number          Description
 S
 S
020 S1AT *+           |      E.FILES    E.NODFIL        File does not exist
 S
021 S0AT *+           |      E.PARMS    E.NEGPAR        File address negative
 S
022 S0AT *+           |      E.PARMS    E.BIGPAR        File address too large
 S
023 AT T U+           |      E.PARMS    E.NEGIX         C-list index negative
```

001 AT T U+ S	|	E.PARMS	E.BIGIX	C-list index exceeds full C-list
002 AT T U+	|	E.OPER	E.CAPTY	Type or options bad

032

(DF: CLKS)

```
001 N     +     | Return and reset disk subsystem clocks
  S
  S       •
  S
002 S1T   +     |     No input parameters
  S
  S
003 S1T   +     |     Returned parameters:
  S
  S
004 S1T   +     |          RDAT D: system time
  S
005 S0T   +     |                  swap time
  S
006 S0T   +     |                  disk sys time
  S
  S
007 S1    +     | The cumulative time expended by the disk subsystem since the last
  S
008       +     | "return and reset" action is returned to the caller.
```

```
PAG LIN                    CODE

001 001 +FRONT+
001 002 +SWIDTH71+
001 003 +DEPTH50+
001 004 +LENGTH71+
001 005 +DOUBLE+
001 006 +MARGIN0+
001 012 +MARGIN35+
001 013 -S1AT1-ECS File Actions -U-Disk File Actions
001 014 -S1AT1-Create a file -U-Create a file
001 015 -AT1-Create a data block -U-Create a data block
001 016 -AT1-Delete a data block -U-Delete a data block
001 017 -AT1-Delete a file -U-Delete a file
001 018 -AT1-Check missing block (probe) -U-Check for missing block on disk file
001 020 -AT1-Read file shape -U-Open read only (read/write)
001 021 -AT1-Read/write information -U-Close
001 022 -AT1-Move a data block -U-Pseudo-close
001 023 -AT1-Test and reset dirty bit -U-Attach file blocks
002 004 +MARGIN0+
002 009 -S1T2-IP1 C: Capability for allocation block (OB.CRFIL)
002 010 -S0T2-IP2 D: C-list index to return capability
002 011 -S0T2-IP3 D: Number of levels in the file
002 012 -S0T2-IP4 D: Pointer to a list of shape numbers
004 002 +MARGIN30+
004 003 -AT1-Class -T3-Number -U-Description
004 004 -S1AT1-E.ABLOCK -T3-E.NOABLK -U-Allocation block does not exist
004 005 -AT1-E.ABLOCK -T3-E.NOECS -U-No ECS available
004 006 -AT1-E.PARMS -T3-E.NEGIX-U-C-list index is negative.
004 007 -AT1-E.PARMS-T3-E.BIGIX-U-C-list index exceeds full C-list
004 008 -AT1-E.PARMS-T3-E.NEGPT-U-Pointer to list of shape numbers is
004 010 -AT1-E.PARMS-T3-E.NEGPAR -U-Level number n < 1
004 011 -AT1-E.PARMS-T3-E.BIGPAR-U-Level number is too large or
004 014 -AT1-E.OPER-T3-E.CAPTY-U-Type or options bad
004 015 -AT1-E.FILES-T3-E.NEGSIZ -U-Non-positive shape number
004 016 -AT1-E.FILES -T3-E.BIGSIZ -U-Shape numbers exceeds 2^17-1
004 017 -AT1-E.FILES -T3-E.NOTPOW-U-Shape number other than S1 not a power of
004 019 -AT1-E.FILES -T3-E.BIGFIL -U-Total size of file exceeds 2^59-1
004 020 +MARGIN0+
005 001 -N-Create a disk file (priviledged operation *)
005 002 -F-* Priviledged operations are used by priviledged system subprocesses
005 003 and are not allowed to be in C-lists.-Z-
005 004
005 005 -S1T1- Input parameters:
005 006 -S0T2-IP1 D: Disk accounting record number
005 007 -S0T2-IP2 BD: Shape numbers S1 through Sn
005 008 -S1T1-Returned parameters:
005 009 -S0T2-RDAT D: Disk unique name, header block size, and disk address
005 010 -S0T3-(one word)
005 011 -S0T2-RCAP C: ECS file capability for new file
006 010 -S1T4-multi-level files - number of levels ≤ 11
006 011 +MARGIN39+
006 013 -S0T11-  512
006 015 -S0T10- ≥ 8
006 016 -S0T10- ≤ 128
006 017 -S0T10- = 2**n for some n
```

```
007 005 +SETTAB 1=3,2=15,3=27+
007 006 -S1T1-Class -T2-Number -T3-Description
007 007 -S1T1-E.PARMS  -T2-E.NEGPAR -T3-Level number < 1
007 008 -S0T1-E.FILES -T2-E.LLEV -T3-Level number too large
007 009 -S0T1-E.FILES -T2-E.NEGSIZ -T3-Data block size negative or (multi-
007 010 -S0T3- level file) too small (< 128)
007 011 -S0T3-Shape number too small (< 8)
007 012 -S0T1-E.FILES -T2-E.BIGSIZ -T3-Data block too big (multi-level > 512) (one-
007 013 -S0T3-level > 508; shape number too big (> 128)
007 014 -S0T1-E.FILES -T2-E.NOTPOW -T3-Data block size or pointer block size
007 015 -S0T3-  (multi-level file) not power of 2
007 016 -S0T1-E.FILES -T2-E.BIGFIL -T3-File too big (> 2^36 - 1)
007 017 -S0T1-E.FILES -T2-E.NOLFH -T3-Too many locally open files
```

```
PAG LIN                    CODE

007 018 -S0T1-E.FILES -T2-E.FULL -T3-Disk system tables full
007 019 -S0T1-E.ABLOCK -T2-E.NOABLK -T3-Disk accounting record does not exist
007 020 -S0T1-E.ABLOCK -T2-E.NOECS -T3-Not enough fixed ECS for file
007 021 -S0T1-E.ABLOCK -T2-E.NODSK -T3-Not enough permanent disk space in
007 022 -S0T3- disk accounting record
007 023 +SETTAB+
008 002 -S1T1-IP1 C: Capability for file (OB.CREBL)
008 003 -S0T1-IP2 D: Address of block in the file
009 002 -S1T1-Class -T3-Number -T5-Description
009 003 -S1T1-E.PARMS -T3-E.NEGIX -T5-C-list index is negative
009 004 -S0T1-E.PARMS -T3-E.BIGIX -T5-C-list index eceeds full C-list
009 005 -S0T1-E.PARMS -T3-E.NEGPT -T5-Address of new block is negative
009 006 -S0T1-E.PARMS-T3-E.BIGPT-T5-Address of new block ≥ file length
009 007 -S0T5- (address range: 0 to length-1)
009 008 -S0T1-E.OPER -T3-E.CAPTY -T5-Type or options bad
009 009 -S0T1-E.FILES -T3-E.NOFIL -T5-File does not exist
009 010 -S0T1-E.FILES-T3-E.ISBLK-T5-Address of new block indicates an already
009 011 -S0T5- existing block
009 012 -S0T1-E.ABLOCK -T3-E.NOECS-T5-No ECS space available
010 002 -S1T1-IP1 C: ECS file capability for locally open disk file (OB.DCRBL)
010 003 -S0T1-IP2 D: Starting file address
010 004 -S0T1-IP3 D: Word count
011 004 +SETTAB1=3,2=12,3=25+
011 005 -S1T1-Class -T2-Number -T3-Description
011 006 -S1T1-E.PARMS  -T2-E.NEGPAR -T3-Nclass -T2-Number -T3-Description
011 007 -S1T1-E.PARMS  -T2-E.NEGIX -T3-C-list index is negative
011 008 -S0T1-E.PARMS -T2-E.BIGIX -T3-C-list index exceeds full C-list
011 009 -S0T1-E.PARMS -T2-E.NEGPAR  -T3-Negative starting file address or negative
011 010 -S0T3- word count
011 011 -S0T1-E.PARMS  -T2-E.BIGPAR -T3-File address plus word count exceeds file
011 012 -S0T3- length
011 013 -S0T1-E.OPER -T2-E.CAPTY -T3-Type or options bad
011 014 -S0T1-E.FILES -T2-E.NODFIL -T3-No such disk file opened by this process
011 015 -S0T1-E.FILES -T2-E.DIOERR¹ -T3-Error on pointer block read from disk
011 016 -S0T3- (modifier = file address of first block not
011 017 -S0T3- created)
011 017 -S0T3- created)
011 018 -F-¹ Error may occur after zero, one or more blocks have been created.-Z-
011 019
011 020 -S0T1-E.FILES -T2-E.ISBLK¹ -T3-Data block address already exists (modifier =
011 021 -S0T3- file address of block which exists)
011 022 -S0T1-E.FILES -T2-E.NABR¹ -T3-No attached block record, i.e., disk system
011 023 -S0T3- out of local storage (modifier = file address
011 024 -S0T3- of first block not created)
011 025 -S0T1-E.ABLOCK -T2-E.NODSK -T3-Insufficient disk space to fund all data blocks
012 001 -S0T1-E.ABLOCK -T2-E.NOSWP -T3-Insufficient swapped ECS in process disk
012 002 -S0T3-accounting record.
012 003 +SETTAB+
013 002 -S1T1-IP1 C: Capability for file (OB.DELBL)
013 003 -S0T1-IP2 D: Address of block to be deleted
013 010 +SETTAB1=3,2=15,3=27+
013 011 -S0T1-Class -T2-Number -T3-Description
013 012 -S1T1-E.PARMS -T2-E.NEGIX -T3-C-list is negative
013 013 -S0T1-E.PARMS -T2-E.BIGIX -T3-C-list index exceeds full C-list
```

```
013 014 -SOT1-E.PARMS  -T2-E.NEGPAR -T3-Pointer is negative
013 015 -SOT1-E.PARMS  -T2-E.BIGPAR -T3-Pointer is too large
013 016 -SOT1-E.OPER -T2-E.CAPTY -T3-Type or options bad
013 017 -SOT1-E.FILES -T2-E.NOBLK -T3-Block to be deleted does not exist
013 018 -SOT1-E.FILES -T2-E.INMAPS -T3-Block to be deleted is in a map
013 019 +SETTAB+
014 002 -S1T1-IP1 C: ECS file capability for locally open disk file (OB.DPLBL)
014 003 -SOT1-IP2 D: Starting address in file
014 004 -SOT1-IP3 D: Word count
015 003 +MARGIN30+
015 004 -S1AT1-Class -T3-Number -U-Description
015 005 -S1T1-E.PARMS -T3-E.NEGIX -U-C-list index is negative
015 006 -SOT1-E.PARMS -T3-E.BIGIX-U-C-list index exceeds full C-list
015 007 -AT1-E.PARMS -T3-E.NEGPAR -U-Negative starting file address or nega
```

```
PAG LIN                    CODE

015 009 -AT1-E.PARMS -T3-E.BIGPAR -U-File address plus word count exceeds file
015 011 -AT1-E.OPER -T3-E.CAPTY -U-Type or options bad
015 012 -AT1-E.FILES -T3-E.NODFIL -U-No such disk file open by calling process
015 013 -AT1-E.FILES -T3-E.DIOERR² -U-I/O error reading pointer block (modifier
015 016 +MARGIN0+
015 017 -F-² Error may occur after zero, one or more block have been deleted.-Z-
015 018
015 019 +MARGIN30+
015 020 -AT1-E.FILES -T3-E.NOBLK² -U-Block to be deleted does not exist (modi
015 023 -AT1-E.FILES -T3-E.TMGA² -U-Block is attached to another process
015 026 -AT1-E.FILES -T3-E.INMAPS²-U-Block in some map in calling process
016 002 +MARGIN0+
017 002 -S1T1-IP1 C: Capability for ECS file (OB.DSTRY)
017 007 -S1T1-Class-T3-Number -T5-Description
017 008 -S1T1-E.PARMS-T3-E.NEGIX-T5-C-list indlist
017 009 -S0T1-E.FILES-T3-E.NOFIL-T5-File to be deleted does not exist
017 010 -S0T1-E.FILES-T3-E.NOTEMP-T5-File to be deleted is not empty
017 011 -S0T1-E.OPER-T3-E.CAPTY-T5-Type or options bad
018 002 -S1T1-IP1 C: Capability for source file (OB.FDFIL, OB.DELBL)
018 003 -S0T1-IP2 D: Address in source file of block to be moved
018 004 -S0T1-IP3 C: Capability for destination file (OB.WFILE, OB.CREBL)
018 005 -S0T1-IP4 D: Address in destination file of block to be moved
018 017 +MARGIN30+
018 018 -S1AT1-E.FILES-T3-E.NOBLK-U-Block to be moved does not exist in
018 020 -AT1-E.FILES -T3-E.ISBLK-U-A block already exists at the designated
018 022 -AT1-E.FILES-T3-E.MISMCH-U-Files do not have equal data block sizes
018 023 -AT1-E.FILES -T3-E.INMAPS-U-Block to be moved is in a map
018 024 -AT1-E.PARMS -T3-E.NEGPT-U-File address negative
019 001 -AT1-E.PARMS-T3-E.BIGPT-U-File address too large
019 002 -AT1-E.PARMS-T3-E.NEGIX-U-C-list index is negative
019 003 -AT1-E.PARMS-T3-E.BIGIX-U-C-list index exceeds full C-list
019 004 -AT1-E.OPER-T3-E.CAPTY-U-Type or options bad
019 005 +MARGIN0+
020 002 -S1T1-Input parameter:
020 003 -S0T2-IP1 C: Disk file capability (OB.OPEN (OB.WFILE))
020 004 -S1T1-Returned parameter:
020 005 -S0T2-RCAP C: ECS file capability for open disk file
022 002 +MARGIN35+
022 003 -S1AT1-OB.DSTRY -U-Destroy a file
022 004 -S0AT1-OB.CHNAM -U-Change unique name
022 005 -S0AT1-OB.CREBL -U-Create a block
022 006 -S0AT1-OB.DELBL -U-Delete a block
022 007 -S0AT1-OB.FDAE -U-Direct ECS access
022 008 -AT1-OB.PLMAP-U-Place in subprocess map
022 009 -AT1-OB.TRDB-U-Test and reset dirty bit
022 010 +MARGIN0+
022 012 +MARGIN35+
022 013 -S0AT1-Class -T3-Number -U-Description
022 014 -S1AT1-E.PARMS -T3-E.NEGIX-U-C-list is negative
022 015 -S0AT1-E.PARMS -T3-E.BIGIX-U-C-list index exceeds full C-list
022 016 -AT1-E.FILES -T3-E.NODFIL -U-No such disk file or
022 018 -S0AT1-E.FILES -T3-E.TMOPN -U-Too many opens (local open count >
022 020 -S0AT1-E.FILES -T3-E.NOLFH -U-Too many locally open files (> ?)
```

```
022 023 -S0AT1-E.ABLOCK -T3-F.NOECS -U-Insufficient fixed ECS to open file
022 024 +MARGIN0+
023 002 -S1T1- IP1 C: ECS file capability (or disk file capability
023 003 -S0T2-for locally open disk file (OB.CLOSE)
024 008 -S1T1-Class -T3-Number -T6-Description
024 009 -S1T1-E.PARMS -T3-E.NEGIX -T6-C-list index is negative
024 010 -S0T1-E.PARMS -T3-E.BIGIX -T6-C-list index exceeds full C-list
024 011 -S0T1-E.OPER-T3-E.CAPTY-T6-Type or options bad
024 012 -S0T1-E.FILES -T3-E.NODFIL -T6-No such locally open file
024 013 -S0T1-E.FILES -T3-E.INMAPS    -T6-Attached block in a map (modifier =
024 014 -S0T6- file address of block in map)
025 002 -S1T1-IPU C: ECS file capability for a locally open disk file (?)
025 003 -S0T1-IP2 D: Starting address in file
```

```
PAG LIN                CODE

025 004 -SOT1-IP3 D: Word count
026 006 +MARGIN30+
026 007 -AT1-Class -T3-Number -U-Description
026 008 -S1AT1-E.PARMS -T3-E.NIGIX -U-C-list index is negative
026 009 -AT1-E.PARMS -T3-E.BIGIX -U-C- list index exceeds full C-list
026 010 -AT1-E.PARMS -T3-E.NEGPAR -U-Negative starting file address or nega
026 012 -AT1-E.PARMS -T3-E.BIGPAR -U-Starting file address plus word count too
026 014 -AT1-E.OPER -T3-E.CAPTY -U-Type or options bad
026 015 -AT1-E.FILES -T3-E.NABR³ -U-No attached block records (disk system
026 018 +MARGIN0+
026 019 -F-0o May occur after zero, one, or more blocks are attached.-Z-
026 020
026 021 +MARGIN30+
026 022 -AT1-E.FILES -T3-E.TMA³ -U-Too many local attaches (> 2¹² - 1)
026 025 -AT1-E.FILES -T3-E.TMGA³ -U-Too many global attaches (> 208 - 1)
026 026 -AT1-E.FILES -T3-E.DIOERR³ -U-Disk I/O error (modifier = file address
027 001 -AT1-E.ABLOCK -T3-E.NOSWP -U-Insufficient swapped ECS in process disk
027 003 +MARGIN0+
028 002 -S1T1-IP1 C: ECS file capability for a locally
028 003 -SOT2-open disk file (OB.    )
028 004 -SOT1-UP2 D: Starting file address
028 005 -SOT1-IP3 D: Word count
028 024 +MARGIN30+
029 001 -AT1-Class -T3-Number -U-Description
029 002 -AT1-E.PARMS-T3-E.NEGIX -U-C-list index is negative
029 003 -AT1-E.PARMS -T3-E.BIGIX -U-C-list index exceeds full C-list
029 004 -AT1-E.PARMS -T3-E.NEGPAR -U-Negative starting file address or
029 006 -AT1-E.PARMS -T3-E.BIGPAR -U-Starting file address plus word count too
029 008 -AT1-E.OPER -T3-E.CAPTY -U-Type or options bad
029 009 -AT1-E.FILES -T3-E.NATH -U-Block not locally attached (modifier =
029 011 -AT1-E.FILES -T3-E.DIOERR -U-Disk I/O error on block (modifier = file
029 013 -AT1-E.FILES -T3-E.TMD -U-Too many detaches (block is a map and
029 016 +MARGIN0+
030 002 -S1I5J10-IP1 C: ECS file capability for primaril file (OB.DERBL, OB.PDLBL)
030 003 -SOI5J10-IP2 C: ECS file capability for secondary file (OB.DLRBL,
031 005 -S1-Possible errors while interchanging file contents: -1T1-Class -T3-Number -T5-Description
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>

          T360.182  INVALID EDIT CODE
031 005 -S1-Possible errors while interchanging file contents: -1T1-Class -T3-Number -T5-Description
<*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*><*>
031 006 +MARGIN25+
031 007 -S1T1-E.OPER -T3-E.CAPTY -U-Type or options bad
031 008 -AT1-E.PARMS -T3-E.NEGIX -U-C-list index is negative
031 009 -AT1-E.PARMS -T3-E.BIGIX -U-C-list index exceeds full C-list
031 010 -AT1-E.FILES -T3-E.NODFIL -U-File not locally open disk file (modifier=1 if
031 012 -AT1-E.FILES -T3-E.INMAPS -U-Block in a map (modifier=file address of
031 014 -AT1-E.FILES -T3-E.TMOPN -U-File is open in another process (modifier=1 if
031 016 -AT1-E.FILES -T3-E.ZLEV -U-File is one level file (modifier=1 if secon
031 018 -AT1-E.ALLOC -T3-E.NODSK -U-Insufficient disk space in file accounting
032 002 -S1T1-IP1 C: Capability for file (OB.RDFIL,(OB.WFILE))
032 003 -SOT1-IP2 D: Address in file
032 004 -SOT1-IP3 D: Address in Central Memory
032 005 -SOT1-IP4 D: Count of words to be transferred
```

```
033 011 +MARGIN30+
033 012 -S1AT1-Class -T3-Number -U-Description
033 013 -S1AT1-E.PARMS -T3-E.NEGIX -U-C-list index is negative
033 014 -AT1-E.PARMS -T3-E.BIGIX -U-C-list index exceeds full C-list
033 015 -AT1-E.PARMS  -T3-E.NEGPAR -U-Word count negative
033 016 -S0AT1-E.PARMS  -T3-E.NEGPT -U-File address negative or
033 018 -S0AT1-E.PARMS  -T3-E.BIGPAR -U-File address plus word count (IP2 + IP4)
033 022 -AT1-E.OPER-T3-E.CAPTY -U-Type or options bad
033 023 +MARGIN0+
034 002 -S1I5J10-IP1 C: ECS file capability for a locally open disk file (OB.ECLM)
034 019 +MARGIN30+
034 020 -S1AT1-Class -T3-Number -U-Description
034 021 -S1AT1-E.PARMS  -T3-E.NEGIX -U-C-list index negative
034 022 -S0AT1-E.PARMS-T3-E.BIGIX -U-C-list index too large
034 023 -S0AT1-E.OPER -T3-E.CAPTY -U-Type or options bad
```

```
PAG LIN                    CODE

034 024 -S0AT1-E.FILES -T3-E.NOFIL -U-No such open disk file
035 001 -S0AT1-E.FILES -T3-E.EXCLAM -U-Local exclusive claim already
035 002 -S0AT1-E.FILES -T3-E.SHCLAM -U-Local shared claim already
035 003 -S0AT1-E.FILES -T3-E.TIMOUT -U-Time out on claim wait queue
035 004 +MARGIN0+
036 002 -S1I5J10-IP1 C: ECS file capability for a locally open disk (OB.SCLM)
036 010 +MARGIN30+
036 011 -S1AT1-Class -T3-Number -U-Description
036 012 -S1AT1-E.PARMS  -T3-E.NEGIX -U-C-list index negative
036 013 -S0AT1-E.PARMS-T3-E.BIGIX -U-C-list index too large
036 014 -S0AT1-R.OPER -T3-E.CAPTY -U-Type or options bad
036 015 -S0AT1-E.FILES -T3-E.NODFIL-U-No such open disk file
036 016 -S0AT1-E.FILES -T3-E.EXCLAM                    -U-Local exclusive claim already
036 017 -S0AT1-E.FILES -T3-E.SHCLAM -U-Local shared claim already
036 018 -S0AT1-E.FILES -T3-E.TIMOUT -U-Time out on claim wait queue
036 019 +MARGIN0+
037 002 -S1I5J10-IP1 C: ECS file capability for a locally open disk file (OB.REL)
037 010 +MARGIN30+
037 011 -S1AT1-Class -T3-Number -U-Description
037 012 -S1AT1-E.PARMS  -T3-E.NEGIX -U-C-list index negative
037 013 -S0AT1-E.PARMS  -T3-E.BIGIX -U-C-list index too large
037 014 -S0AT1-E.OPER -T3-E.CAPTY -U-Type or options bad
037 015 -S0AT1-E.FILES -T3-E.NODFIL-U-No such open disk file
037 016 -S0AT1-E.FILES -T3-E.NOCLAM -U-No local claim exists
037 017 +MARGIN0+
038 002 -S1T1-IP1 C: Capability for file
038 003 -S0T1-IP2 D: Address of buffer for the shape numbers
038 004 -S0T1-IP3 D: Buffer size
038 016 +MARGIN30+
038 017 -S1AT1-Class -T3-Number -U-Description
038 018 -S1AT1-E.FILES -T3-E.NOFIL  -U-File whose shape is to be read does not
038 020 -AT1-E.PARMS -T3-E.NEGIX -U-C-list index negative
038 021 -AT1-E.PARMS -T3-E.BIGIX -U-Coslist index exceeds full C-list
038 022 -AT1-E.PARMS  -T3-E.NEGPT -U-Buffer address is negative
038 023 -S0AT1-E.PARMS  -T3-E.NEGPAR -U-Buffer size ≤ 0
038 024 -S0AT1-E.PARMS  -T3-E.BIGPAR -U-Buffer address plus size exceeds user's
039 002 -AT1-E.OPER -T3-E.CAPTY -U-Type or options bad
039 003 +MARGIN0+
040 002 -S1T1-IP1 C: Capability for ECS file
040 003 -S0T1-IP2 D: Address of block in ECS file
040 013 +MARGIN30+
040 014 -S1AT1-Class -T3-Number -U-Description
040 015 -S1AT1-E.FILES -T3-E.NODFIL-U-The file does not exist
040 016 -AT1-E.OPER -T3-E.CAPTY-U-Type or options bad
040 017 -AT1-E.PARMS -T3-E.NEGIX -U-C-list index is negative
040 018 -AT1-E.PARMS -T3-E.BIGIX -U-C-list index exceeds full C-list
040 019 -AT1-E.PARMS -T3-E.NEGPAR-U-The address of the block is negative
040 020 -AT1-E.PARMS-T3-E.BIGPAR-U-The address of the block is too large
040 021 +MARGIN0+
041 002 -S1AT1-No input parameter
041 009 +MARGIN30+
041 010 -AT1-E.FILES -T3-E.INMAPS-U-Some block of some file is in a map (file
041 012 +MARGIN0+
042 002 -S1T1-Input parameters:
```

```
042 004 -S0I10J15-IP2 D: Zero for reset; non-zero for
042 005 -S1T1-Returned parameter
042 006 -S1I10J15-X6 is set to the old value of the over-ride flag
042 014 +MARGIN30+
042 015 -AT1-Class -T3-Number -U-Description
042 016 -AT1-E.FILES -T3-E.NODFIL -U-No such locally open disk file
042 017 -AT1-E.PARMS -T3-E.NIGIX -U-C-list index is negative
042 018 -AT1-E.PARMS -T3-E.BIGIX -U-C-list index exceeds full C-list
042 019 -AT1-E.OPER -T3-E.CAPTY -U-Type or options bad
042 020 +MARGIN0+
043 002 -S1I5J10-IP1 C: ECS file capability for a locally open disk file
043 003 -S0I5J10-IP2 D: Address of a block in the file
043 014 +MARGIN30+
```