1/21/70

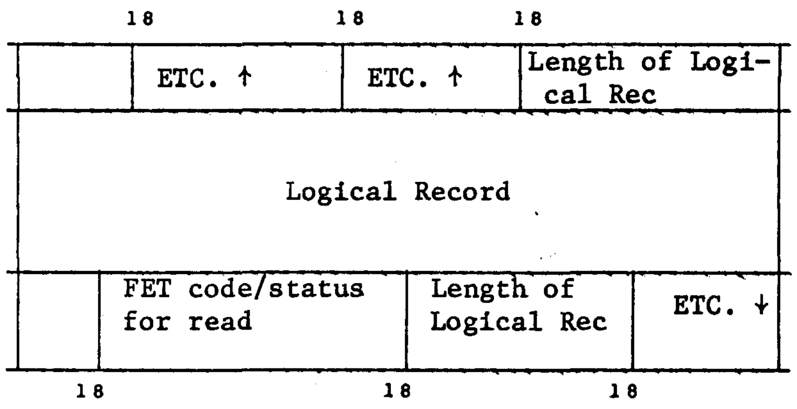
SCOPE Manual

This document describes the SCOPE simulator under the interim system. A knowledge of the SCOPE 3.1 manual is assumed.

1.0 SCOPE

1.1 There are three main sections of the simulator: the command processor; the file activity simulator; the loader. The command processor gives the user control over the activities of the system. The SCOPE activity simulator processes the RA+1 requests of the program. It also handles file positioning for the command processor. The loader handles relocatable file loading for the command processor. It also produces load maps and absolute overlays.

1.2 There are three types of files in the interim system: System standard text (SYSTEXT) files; SCOPE simulator files; core dump files. SYSTEXT files are described in another document. SCOPE simulator files consist of logical records blocked into TSS files in the following format:



Core dump files are absolute core images used for programs in the interim system. All of the processors are in this format. They are generated by the loader in the overlay operation.

1.3 SCOPE maintains a moderate sized local directory used to hold entries from the master directory (see the BEAD document). Files not in the local directory and requested by the SCOPE activity simulator are automatically requested from the BEAD with the local user name and put into the local directory rewound.

2.0 The COMMAND Processor

2.1 The syntax for typing to SCOPE is as follows:

```
<command> ::= [<word><separator>] ...
<word> ::= a string of characters except '='. '.'. '(', ')',   CR
<separator>  ::= '=', ',', '(', ')',   CR
```

2.2 SCOPE file names contain 7 characters. File specifiers in the interim system are 15 characters divided 7 for the file name and 8 for user name. The SCOPE file name is the same as the interim system file name. It is not possible to have two files with the same name but different user names in SCOPE.

2.3 Commands:

2.3.1 TIME,ΔHH.MM.SS.CR

The time is entered into the simulator.

2.3.2 DATE,ΔMM/DD/YY CR

The date is entered into the simulator.

2.3.3 RFL,n

The field length is set to n $\leq$ 50000B.

2.3.4 ZERO

The field length is zeroed.

2.3.5 L,fname

The file *fname* is loaded into core. The simulator enters loading mode.

2.3.6 GO,entry name, parameters

This command acts the same as the SCOPE 3.1 execute card. It is used after loading a set of files into memory.

2.3.7 OVERLAY,*fname*,*user-name*

This produces a core dump file onto *fname*, *user-name*, which is obtained and returned directly from the BEAD

2.3.8 X,*library program*, parameters

The file *library-program* with user name *LIBRARY* is loaded (must be overlay load) and entered with given parameters. The library-program-file is returned to the BEAD.

2.3.9 GET,*fname,user-name*,mode,read-only

This command is used to obtain files that do not have your user name. (Ones that do are obtained and rewound automatically.) Mode is either TEXT or SCOPE if it is a new file. If this parameter is omitted and it is a new file, the command is ignored with an appropriate message. Read only is used if the file is to be read only. Set the field non-null to specify read/only.

2.3.10 PUT,*fname*

The file is returned to the BEAD.

2.3.11 FILE,*fname,user-name*

The specified SYSTEXT file is read until end of information and treated as commands.

2.3.12 MESSAGES,$^{\text{ON}}_{\text{OFF}}$

Turns on/off messages from the program.

2.3.13 DEBUG

Calls the BEAD to enter BEAD commands.

2.3.14 FORGET,*fname*

The file name is deleted from the local directory but not returned to the BEAD.

2.3.15 ECS

Makes the ECS core file with120000B words. This file is available to the user through a C-list index given in B1 at transfer time.

2.3.16 DONE

This command is used to leave the SCOPE simulator and return to the BEAD. The non-read/only files in the local directory are rewound and returned to the BEAD.

2.3.17 COMLENTH=n

This command allocates n words for common blocks during loading. If n is too small, TOO BIG will appear when a load is attempted.

2.3.18 SPACE=n

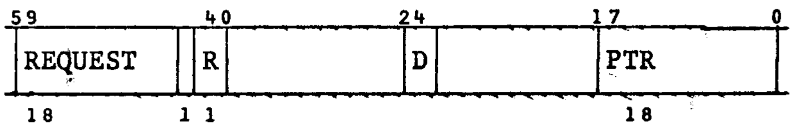
This command sets the limit on the amount of file space that the simulator will create for files.

2.3.19 RESTORE

When SPACE EXHAUSTED is typed, an additional length of file space may be entered and RESTORE to resume program execution.

3.0 SCOPE Activity Simulator

3.1 This section is responsible for the processing of user requests over the standard SCOPE interface of 'RA+1'. The REQUEST word is interpreted as follows:



The REQUEST field is generally interpreted as 3-display-coded characters. The PTR field is a pointer to parameters for the individual actions.

3.2 REQUESTS' LISTED are the display-coded contents of the REQUEST field.

3.2.1 CIO

The PTR field contains a pointer to an FET.

3.2.1.1 Maintained by CIO is a REUQEST QUEUE. All REQUESTS with an R field of zero are queued instead of completed. The queue is processed on an RCL, END, or ABT request.

3.2.1.2 The FET is modeled after the SCOPE 3.1 FET. The words and fields recognized by the simulator are described below

42	18
Logical File Name	Code/Status
	FIRST
0	IN
0	OUT
64_{10}	LIMIT
18	18

3.2.1.3 The action to be taken by CIO are passed in the Code/Status field. The binary/coded bit is ignored.

3.2.1.3.1 READ (10B)

The buffer is filled with information from the file. A SYSTEXT to DISPLAY Code conversion is performed if the file is a SYSTEXT file. If the end of the logical record is reached, the status field is set to the value of the status field when the file was written without the low order '4' digit of the WRITE of code. If a SYSTEXT EOI word is encountered, the status field is set to 741033B.

3.2.1.3.2 WRITE (X4B) X = 1,2,3.

The buffer is emptied onto the file. A DISPLAY Code to SYSTEXT conversion is performed if the file is a SYSTEXT file. If X is 2 or 3 a logical record control word is written using the status field. X is considered part of the status field.

3.2.1.3.3 BACKSPACE ($4Y_8$) Y is ignored.

This is equivalent to a SKIP with RECORD count of -1.

3.2.1.3.4 REWIND (XY_8) x = 5,6. Y is ignored.

If the last operation on the file was WRITE, an End-of-information is placed on the file. The file is rewound. An error is produced if the file was writing and a logical record is not closed.

3.2.1.3.5 EVICT ($14Y_8$) Y is ignored.

The file is deleted from the system through B2 releasing the space used by the file.

3.2.2 OPE

The PTR field contains a pointer to an FET. The 64_{10} field of the FET is set to 64_{10}.

3.2.3 TIM

The PTR field points to a reply cell. If the D field contains a 1, the DATE is returned in the reply field. Else the TIME is returned.

3.2.4 LDR

The PTR field contains a pointer to a two-word parameter package. The format of this area is given in the SCOPE 3.1 manual. The specified file to be loaded must be in overlay format.

3.2.5 END-ABT

The PTR field is ignored. END or ~~ABT~~ ABORT is typed on the teletype. The REQUEST queue is dumped. The command processor is entered.

3.2.6 RCL

The PTR field is ignored. The REQUEST queue is dumped.

3.2.7 LIB

The PTR field points to a word containing an EXTERNAL NAME in display code left-justified with zero fill.

3.2.7.1 A special file in the system called DIRECT,LIBRARY contains a list of externals correlated with the file name they are defined on. Each list entry is two words: 1) external name; 2) file name, both in display code, left-justified with zero fill. The user name for these file names is LIBRARY.

3.2.7.2 LIB locates the external in the library directory. The file name associated with the external is returned in the REQUEST word. The file name with user name LIBRARY is obtained by calling the BEAD. LIB is primarily used by the SCOPE Loader.

3.2.8 MSG

The PTR field is as in SCOPE 3.1 MSG. If the message switch is on, the message is typed onto the teletype.

3.2.9 GSM

The PTR field points to an array orea that is to receive a message typed in from the teletype. The message is placed one character per word RIGHT justified display code. It is ended by a word of all zeros.

An up arrow is typed on the teletype. Characters typed are accepted and placed into the array until the carriage return. The standard line editor features are available.

3.2.10 MEM

The pointer field and actions taken are described in the SCOPE 3.1 manual.

4.0 The Loader

4.1 The loader operation is divided into three areas: 1) initialization; 2) file loading; 3) wrapup and transfer. The first call by the command processor to load a file causes both the loading of that file plus initialization. Subsequent calls during the same loading sequence load the specified files. Finally, the command processor calls to transfer on a GO command, or, to produce a core dump file, on an OVERLAY command.

4.2 Initialization involves setting up the tables and allocating words needed by the loader. All of this information is kept at the high end of core.

4.2.1 The address to load the next program, the program origin call, is initialized at 100_8.

4.2.2 The address to load the next COMMON Block is initialized to the field length minus the last COMBKLEN command entered.

4.2.3 The entry name/COMMON Block name table is initialized to contain blank COMMON.

4.2.4 The highest length request for blank COMMON is initialized to zero.

4.3 Each file to be loaded is first rewound. The '77' table is ignored.

4.3.1 A maximum of 500 references to different COMMON blocks is permitted.

4.3.2 References to BLANK COMMON core converted to references to the external *BLNK C, and a maximum length request is maintained.

4.3.3 If an absolute origin exceeds the contents of the program origin cell, the program origin is set to the absolute origin + 1.

4.3.4 REPL tables are processed when encountered.

4.4 WRAPUP and TRANSFER initials a series of operations.

4.4.1 UNSATISFIED Externals are checked for difinition in the LIBRARY directory using the LIB routine. File names returned by LIB are loaded. This step repats until either there are no further unsatisfied externals or none of the externals are found in the Library Directory,

4.4.2 If BLANK COMMON was requested, it is defined by the final contents of the program origina cell. Note that any COMMON BLOCKS appear after BLANK COMMON.

4.4.3 A Load map is produced on the file OUTPUT. Entry names are listed alphabetically sorted.

4.4.4 If a transfer is to occur, the last word loaded address is placed in cell 65_8 and an image of the GO command suffixed by a period is placed in cells 70_8-77_8. The individual parameters are placed in cell 2. The parameter count is put in cell 67_8. The program is entered at the entry name given on the GO command; or if that field is null, it is entered according to the last transfer table encountered.

4.4.5 If an overlay is to be made, core is written out onto the given file starting at location zero and continuing through the last word loaded.

5.0 SCOPE Type Outs

5.1 Most type outs made by the SCOPE simulator are listed here. They have been divided according to the section of the SCOPE simulator that produces them. Other type outs are considered self-explanatory and are not listed.

5.2 COMMAND Processor Type Outs

5.2.1 MORE THAN 20 PARAMETERS

A command contains more than 20 parameters. The command was ignored.

5.2.2 ? ? ?

The first word of the command was not in the list of commands taken. The command was ignored.

5.2.3 PAR #>10 CHARS

A parameter in a command exceeded 10 characters. The command was ignored.

5.2.4 NOT FOUND

The file name in a PUT or FORGET command could not be located in the local directory.

5.2.5 NEW FILE .. GIVE MODE

The first file block in the file delivered by the BEAD during a GET command does not exist and no mode was specified in the command. A PUT was done on the file descriptor to return the file to the BEAD. The GET will have to be re-entered with a mode specified.

5.2.6 WILL CAUSE DUPLICATE FILES

The file name specified for a GET command already appears in the directory. The USER names of the two files are irrelevant. The command was ignored.

5.2.7 OBTAINED

The file specified in the GET command was a file that already existed in the system.

5.2.8 GENERATED

The file specified in the GET command was initialized in the appropriate mode to a null file.

5.2.9 SPACE EXHAUSTED

The cell set by the space command has gone to zero due to the file blocks that have been created since the SPACE command was entered. A new space figure may be entered and the RESTORE command used to resume operation.

5.3 SCOPE Activity Simulator Type Outs

All type outs are fatal and cause the command processor to be entered.

5.3.1 PP CALL ERROR

The REQUEST field of the RA+1 request contains an unknown configuration.

5.3.2 REQ PTR $\overset{>}{\nmid}$ FL

The PTR field of the RA+1 request is negative or points outside the current field length.

5.3.3 INVALID FET REQUEST

The code/status field contains an unrecognizable quantity.

5.3.4 BAD FET

The buffer pointers (FIRST, IN, OUT, LIMIT) contains an illegal configuration. IN and OUT must be between FIRST and LIMIT. FIRST must be less than LIMIT. FIRST must be positive. LIMIT-1 must lie within the field length.

5.3.5 UNEXPECTED EOI OR NULL FILE

This will happen if an X command is given for a library program that does not exist. All other occurrences spell disaster.

5.3.6 BLOCK DELETED FROM FILE

This spells disaster.

5.3.7 READ REQUEST ILLEGAL WITH CURRENT SECURITY CODE

This will happen if a READ after WRITE is attempted or if a READ at end-of-information is attempted.

5.3.8 ATTEMPT TO WRITE ON READ ONLY FILE

A file that was gotten with the RO field of the GET command non-null may not be written upon.

5.3.9 WRITE REQUEST ILLEGAL WITH CURRENT SECURITY CODE

Writing after reading is NOT permitted except at EOI.

5.3.10 SYSTEXT FILE DOES NOT END WITH CR

An EOI was written or encountered upon reading on a SYSTEXT file whose last line does not end with a carriage return.

5.3.11 REWIND GIVEN ON FILE THAT HAS OPEN LOGICAL RECORD

A rewind was given on a file whose last operation was NOT WRITER or WRITEF, but rather WRITE, thus leaving an open logical record.

5.3.12 MEM REQ > 50000

A mem call to request field length greater than 50000_8 was made.

5.4 LOADER TYPEOUTS

All type outs except 5.4.1 are fatal and cause the command processor to be called.

5.4.1 BLANK COMMON TRUNCATED BY LOADER

The maximum length of BLANK COMMON is too big for the current field length. BLANK COMMON length was truncated and WRAPUP continues.

5.4.2 DUPLICATE PROGRAM IN LOAD FILE

Multiple decks with the same program name are not permitted.

5.4.3 LOADER TABLE ERROR

An unrecognizable table was encountered.

5.4.4 LOAD ADDRESS IS NEGATIVE

An address in a loader table is negative.

5.4.5 TOO BIG

The loader is not able to fit everything into core. Can be caused by COMBKLEN being too small or the current FL too small.

5.4.6 NO ENTRY FOR TRANSFER

The TRANSFER LABEL from either the GO command or the last transfer table cannot be found in the list of entry points.

5.4.7 LOAD OR RELOCATION ATTEMPTED ON BLANK COMMON

It is not permissible to load anything into BLANK COMMON.

6.0 THE LIBRARY

6.1 The contents of files in the LIBRARY are divided into three ~~classes~~ types: 1) overlay format program; 2) relocatable format programs; 3) relocatable routines indexed in the library directory. All have user names of LIBRARY.

6.2 Overlay format files are the only type of file that the LDR request will load or can be run using the X command. They consist of an optional '77' table followed by a '50' table.

6.3 Relocatable format programs must be run using the LOADER section of the simulator. Note that the loader will also ~~type~~ load 1 programs.

6.4 The relocatable format routines of type 3 are loaded automatically by the loader to satisfy externals. They are indexed in a file as described in 3.2.7.1.

6.5 Some of the programs in the library are listed here with their classification.

6.5.1 COPY(2): The standard SCOPE 3.1 copy.

6.5.2 COPYL(2): Standard SCOPE 3.1 copyL.

6.5.3 COMPASS(1): CAL compass

6.5.4 SNOBOL(1): CAL SNOBOL

6.5.5 REWIND(1): Used to rewind a file.

6.5.6 DEBUG(3): CAL DEBUG

6.5.7 UPDATE(2): SCOPE 3.1 UPDATE

6.5.8 CATALOG(2): CAL CATALOG