UTILITIES

This section of the handbook includes documentation on the following software:

CREF	Version 47
FILCOM	Version 20
FUDGE2	Version 15
GLOB	Version 5A

These utilities are used by system programmers

1. To obtain cross-referenced listings for all operand-type symbols, user-defined symbols, and/or op codes and pseudo-op codes.

2. To compare two versions of a file and then output any differences.

3. To update files containing relocatable binary programs and manipulate programs within program files.

4. To obtain an alphabetical cross-reference listing of all global symbols encountered.

With the exception of the CREF writeup, the utilities have been reproduced from the DECsystem-10 Operating System Commands manual (DEC-10-MRDC-D).

CROSS-REFERENCE LISTING (CREF)

CREF produces a sequence-numbered assembly listing followed by one to three tables, one showing cross references for all operand-type symbols (labels, assignments, etc.), another showing cross references for all user-defined operators (macro calls, OPDEFs etc.), and another (if the proper switch is specified) showing the cross references for all op codes and pseudo-op codes (MOVE, XALL, etc.). A number sign (#) appears on the definition line of all symbols. The input to CREF is a modified assembly listing file created during a MACRO-10 assembly or FORTRAN IV compilation when the /C switch is specified in the command string.

CREF provides an invaluable aid for program debugging and modification.

1.0 REQUIREMENTS

Minimum Core:	2K pure, 1K impure
Additional Core:	Takes advantage of any additional core available, as necessary.
Equipment:	One input device (normally disk) which contains the modified assembly listing file; one output device (normally the line printer) for the listing.

2.0 INITIALIZATION

.R CREF↵	Loads the Cross-Reference Listing program into core.
*	The program is ready to receive a command.

NOTE

If CREF cannot initialize the terminal, it exits.

3.0 COMMANDS

3.1 Command Formats

a. output-dev:filename.ext[proj,prog]=input-dev:file1.ext[proj,prog],file2.ext,...

b. progname!

output-dev:filename.ext

The device on which the assembly listing and cross-reference tables are to be printed. If no output file name is specified, the default file name is the same as that specified for the first input file, but with the extension .LST. In such a case, the default device is LPT. However, if an output file name is specified, the default output device is DSK.

input-dev:filename.ext

The device on which the modified assembly listing was written during MACRO-10 assembly. DSK: is assumed if the device is not specified. When looking for the input file, CREF tries the following default extensions in the order listed: .CRF, .LST, .TMP, or a null extension. A missing input file name is given the name CREF. If the input file extension is .CRF or .LST and the /P switch is not included in the command string, the input file is deleted after the output file is successfully closed.

Multiple input files can be specified to be combined into a single CREF listing by separating the input files with commas. Switches affecting the entire listing (/K, /M, /O, and /S) must be specified before the terminator for the first input file. Switches affecting the positioning of an input file are specified with each file.

The ?CANNOT FIND FILE... message will be printed for each occurrence of a missing file. If the missing file is not part of a COMPIL-class command file (that is, if it was typed in directly), the command will be aborted, allowing the user to retype the command string. However, if the missing file is part of a COMPIL-class command file, processing will continue for the rest of the existing files in the command string. (Refer to section 4.0 of this document.) Note that if any file is in fact missing, no input file will be deleted.

[proj,prog]

The disk area on which the files are to be placed (output) or the disk area on which the source files reside (input). If omitted, the default is the user's disk area.

=

The output device and the input device are separated by an equal sign. If the equal sign is omitted, output defaults occur as described above. Any files specified by the user are for input.

progname!

The user can request CREF to run a system program by typing the program name followed by an exclamation point.

Examples of Commands:

```
.R MACRO)
*PTP:=/C=DTA1:TXCALC)
THERE ARE NO ERRORS

PROGRAM BREAK IS 003771
```

Load the MACRO-10 Assembler into core.

Assemble the program TXCALC from DTA1; writes the object program coding on the paper tape punch; writes a modified assembly listing on DSK: (assumed) and assigns it the filename CREF.LST.

```
7K CORE
*↑C
.R CREF)
*)
```

Return to the monitor.

Load CREF into core.

Select the default assumptions of:

output-dev:	LPT:
input-dev:	DSK:
input filename.ext	CREF.CRF (.LST,.TMP)
output filename.ext	CREF.LST

Equivalent to the command string

LPT:CREF.LST=DSK:CREF.CRF

```
*↑C
.
```

Return to the monitor.

```
.R CREF
*OUTFIL=FIL1,FIL2,FIL3
```

Make single merged cross-reference file for three program files.

```
*LINK!
*FIL1,FIL2,FIL3/G
LINK: LOADING
EXIT
.
```

Run LINK10.

3.2 Switches

Switches are used to specify such options as magnetic tape control and list selection. All switches are preceded by a slash (/).

Examples of Switches:

```
.R CREF
*/M=MTA1:/W
```

Load CREF into core.

Rewind MTA1 and process the first file, listing only the cross references for operand-type symbols (labels, assignments, etc.).

```
*DTA5:SAVE1/Z=
```

Process the file named CREF.LST in the user's area of disk; write the program listing and operand-type cross references on DTA5 and call the file SAVE1.

```
*↑C
```

Return to monitor.

CREF Switch Options

Switch	Meaning
A	Advance magnetic tape reel by one file. /A may be repeated.
B	Backspace magnetic tape reel by one file. /B may be repeated.
H	Print help for running CREF.
K	Kill listing of references to basic symbols (labels, assignments, etc.).
M	Suppress listing of references to user-defined operators (Macro calls, OPDEFs, etc.).
O	Allow listing of references to machine and pseudo-operation codes (MOVE, XALL, etc.).
P	Preserve an input file with the extension .CRF or .LST, which is normally deleted.
R	Request the line number at which the listing is to Restart. CREF prints: RESTART LISTING AT LINE: at which time the user types the line number followed by a carriage return. (Such action might be necessary if the line printer ran out of paper, or jammed, etc.)
S	Suppress program listing (list only the selected tables).
T	Skip to logical end of magnetic Tape.
W	ReWind magnetic tape.
Z	Zero the DECtape directory (DECtape must be output only).

4.0 MONITOR COMMANDS

CREF-format listing files generated by COMPILE, LOAD, EXECUTE, and DEBUG commands (using the /CREF switch) can be printed on the line printer by typing

.CREF↵

The CREF command will print out all listing files that are specified in the COMPIL-class command file, nnnCRE.TMP (where nnn is the user's job number). It will also transfer control to a system program if its name is present in the form "progname!". After completion of this operation, nnnCRE.TMP is deleted to prevent the listing files from being listed again by the next CREF command.

The CREF files may also be listed by an R CREF command and a response of "filename" to each asterisk (*) typed by CREF. It is important to note that, if the user uses the R CREF command to list files created by the monitor's COMPIL-class commands, the names of the files to be listed must be typed in response to the asterisks.

5.0 DIAGNOSTIC MESSAGES

CREF Diagnostic Messages

Message	Meaning
?dev NOT AVAILABLE	Device is assigned to another job.
?CANNOT ENTER FILE, n dev: file.ext	DTA or DSK directory is full; file cannot be entered; n indicates the cause of the failure and is obtained from the ENTER directory block.
?CANNOT FIND FILE, n dev: file.ext	The file cannot be found on the device specified; n indicates the cause of the failure and is obtained from the LOOKUP directory block.
?COMMAND ERROR--TYPE /H FOR HELP	Error in last command string entered. 1. Device name, file name, or extension consisted of non-alphanumeric characters. 2. The project-programmer number was not in standard format (i.e., it was not octal numbers in the form [proj, prog]). 3. An undefined switch was specified, switches in parentheses were not separated by commas, or the closing parenthesis was missing. 4. The /Z switch was used on the input side of the command string.
?COMMAND FILE INPUT ERROR, n dev:file.ext	Disk data error while reading nnnCRE.TMP; n is a six-digit (or less) octal number representing the file status word returned from the GETSTS UUO.
?DATA ERROR DEVICE dev:	READ or WRITE error.
IMPROPER INPUT DATA, CONTINUING	Input data not in CREF format. Output listing continues.
?INPUT BUFFERS TOO BIG	The monitor set up input buffers longer than 203_8. This is not a user error and hopefully will never occur.
?INPUT ERROR, n dev:file.ext	READ error has occurred on the device.
?INSUFFICIENT CORE	Additional core is required for execution but none is available from monitor.
?OUTPUT ERROR, n dev:file.ext	WRITE error has occurred on the device.

Function

The FILCOM program is used to compare two versions of a file and to output any differences. Generally, this comparison is line by line for ASCII files or word by word for binary files. FILCOM determines the type of comparison to use by examining either the switches specified in the command string or the extensions of the files. Switches always take precedence over file extensions.

Command Format

```
.R FILCOM
*output dev:file.ext [directory] = input dev1:file.ext [directory],
        input dev2:file.ext [directory]
```

output dev: = the device on which the differences are to be output.

input dev: = the device on which an input file resides.

Defaults

1. If the entire output specification is omitted, the output device is assumed to be TTY. However, the equal sign must be given to separate the input and output specifications of the command string.
2. If an output filename is specified, the default output device is DSK.
3. If the output filename is omitted, the second input filename is used, unless it is null. In this case, the filename FILCOM is used.
4. If the output extension is omitted, .SCM is used on a source compare and .BCM is used on a binary compare.
5. If the [directory] is omitted (input or output side), the user's default directory is assumed.
6. If an input device is omitted, it is assumed to be DSK.
7. If the filename and/or extension of the second input file is omitted, it is taken from the first input file.
8. A dot following the filename of the second input is necessary to explicitly indicate a null extension, if the extension of the first input file is not null. For example, to compare FILE.MAC and FILE. (i.e., with null extension), use the following command string:

```
.R FILCOM
*=FILE.MAC,FILE.
```

Command Format (cont)

9. The second input file specification cannot be null unless a binary compare is being performed. In a binary compare, if the first input file is not followed by a comma and a second input file descriptor, the input file is compared to a zero file and is output in its entirety. This gives the user a method of listing a binary file. Refer to Example 4.

Switches

The following switches can appear in the command string, depending on whether a source compare or a binary source compare is being performed.

Binary Compare

/H	Type list of switches available (help text from device SYS:).
/nL	Specify the lower limit for a partial binary compare (n is an octal number). This switch, when used with the /nU switch, allows a binary file to be compared only within the specified limits.
/Q	When the files are different, print the message ?FILES ARE DIFFERENT, but do not list the differences. This switch is useful when BATCH control files want to test for differences but do not want the log file filled with these differences.
/nU	Specify the upper limit for a partial binary compare (n is an octal number). This switch, when used with the /nL switch, allows a binary file to be compared only within the specified limits.
/W	Compare files in binary mode without expanding the files first (refer to Appendix D). This switch is used to compare two binary files with ASCII extensions.
/X	Expand SAV files before comparing them in binary mode. This action removes differences resulting from zero compression (refer to Appendix D).

Source Compare

/A	Compare files in ASCII mode. This switch is used to force a source compare on two ASCII files.
/B	Compare blank lines. Without this switch, blank lines are ignored.
/C	Ignore comments (all text on a line following a semicolon) and spacing (spaces and tabs). This switch does not cause a line consisting entirely of a comment to become a blank line, which is normally ignored.
/H	Type list of switches available (help text from device SYS:).

(continued on next page)

Command Format (cont)

Source Compare (cont)

/nL Specify the number of lines that determine a match (n is an octal number). A match means that n successive lines in each input file have been found identical. When a match is found, all differences occurring before the match and after the previous match are output. In addition, the first line of the current match is output after the differences to aid in locating the place within each file at which the differences occurred. The default value for n is 3.

/Q Print the message ?FILES ARE DIFFERENT, when the files are different, but do not list the differences.

/S Ignore spaces and tabs.

/U Compare in update mode. This means that the output file consists of the second input file with vertical bars (or back slashes for 64-character printers) next to the lines that differ from the first input file. This feature is useful when updating a document because the changes made to the latest edition are flagged with change bars in the left margin. The latest edition of the document is the second input file.

If switches are not specified in the command string, the files are compared in the mode implied by the extension. The following extensions are recognized as binary and cause a binary compare if one or both of the input files have one of the extensions.

.BAC	.HGH	.RMT
.BIN	.LOW	.RTB
.BUG	.MSB	.SAV
.CAL	.OVR	.SFD
.CHN	.QUE	.SHR
.DAE	.QUF	.SVE
.DCR	.REL	.SYS
.DMP	.RIM	.UFD
		.XPN

Binary files are compared word by word starting at word 0 except for the following two cases:

1. Files with extensions .SHR and .HGH are assumed to be high segment files. Since the word count starts at 400000, upper and lower limits, if used, must be greater than (or equal to in the case of the lower limit) 400000.
2. Files with extensions .SAV, .LOW, and .SVE are assumed to be compressed core image files and are expanded before comparing.

Command Format (cont)

Conflicts are resolved by switches or defaults. If a conflict arises in the absence of switches, the files are assumed to be ordinary binary files.

Output

In most cases, headers consisting of the device, filename, extension, and creation date of each input file are listed before the differences are output. However, headers do not appear on output from the /U switch (update mode on source compare).

Source compare output - After the headers are listed, the following notation appears in the left column of the output

n)m

where

n is the number of the input file, and
m is the page number of the input file (see examples).

The right column lists the differences occurring between matches in the input files. Following the list of differences, a line identical to each file is output for reference purposes.

The output from the /U switch differs from the above-described output in that the output file created is the second input file with vertical bars in the left column next to the lines that are different from the first input file.

Binary compare output - When a difference is encountered between the two input files, a line in the following format appears on the output device:

octal loc. first file-word second file-word XOR of both words

If the exclusive OR (XOR) of the two words differs only in the right half, the third word output is the absolute value of the difference of the two right halves. This usually indicates an address that changed.

If one input file is shorter than the other, after the end of file is encountered on the shorter file, the remainder of the longer file is output.

Characteristics

The R FILCOM command:

Places the terminal in user mode.
Runs the FILCOM program, thereby destroying the user's core image.

Associated Messages

?2K CORE NEEDED AND NOT AVAILABLE

FILCOM needs 2K of core to initialize I/O devices and this core is not available from the monitor.

?BUFFER CAPACITY EXCEEDED AND NO CORE AVAILABLE

The buffer is not large enough to handle the number of lines required for looking ahead for matches, and additional core is not available.

?COMMAND ERROR

One of the following errors occured in the last command string typed.

1) There is no separator (←or =) between the output and input specifications.
2) The input specification is completely null.
3) The two input files are not separated by a comma.
4) A file descriptor consists of characters other than alphanumeric characters.
5) FILCOM does not recognize the specified switch.
6) The project-programmer number is not in standard format, i.e., [proj,prog].
7) The value of the specified switch is not octal.
8) The first input file is followed by a comma but the second input file is null.

?DEVICE dev: NOT AVAILABLE

Device is assigned to another job or does not exist.

?FILE n NOT IN SAV FORMAT

The user indicated via the /X switch that the file is to be expanded but the specified file is not in compressed file format. N is either 1 or 2 indicating the first file or the second file.

Associated Messages (Cont)

?FILE n READ ERROR

An error has occurred on either the first or second input device.

%FILES ARE DIFFERENT

The two input files specified in the command string are different (i.e., the two files are not two versions of the same file but are two different files).

?INPUT ERROR - file.ext FILE NOT FOUND

The specified file could not be found on the input device.

NO DIFFERENCES ENCOUNTERED

No differences were found between the two input files.

?OUTPUT DEVICE ERROR

An error has occurred on the output device.

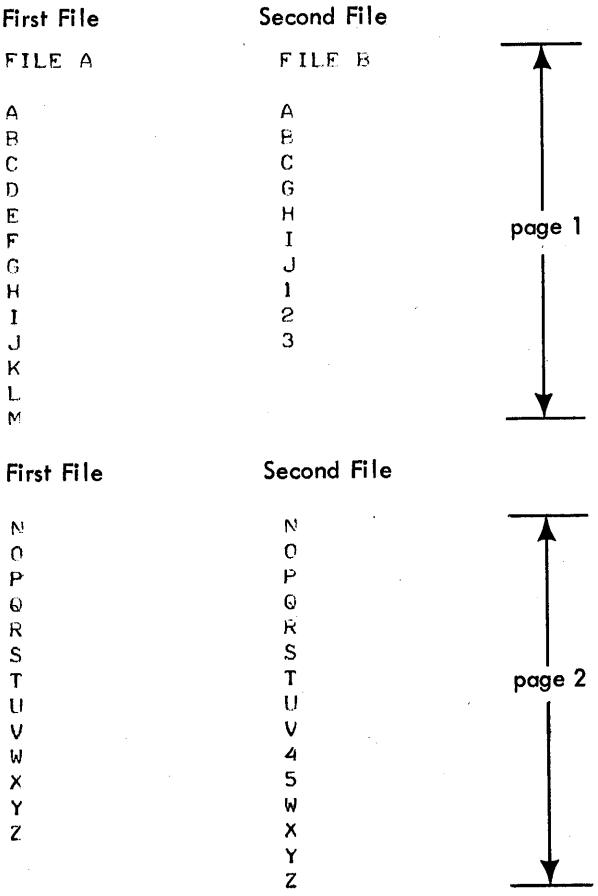
?OUTPUT INITIALIZATION ERROR

The output device cannot be initialized for one of the following reasons:

1) The device does not exist or is assigned to another job.
2) The device is not an output device.
3) The file cannot be placed on the output device.

Examples

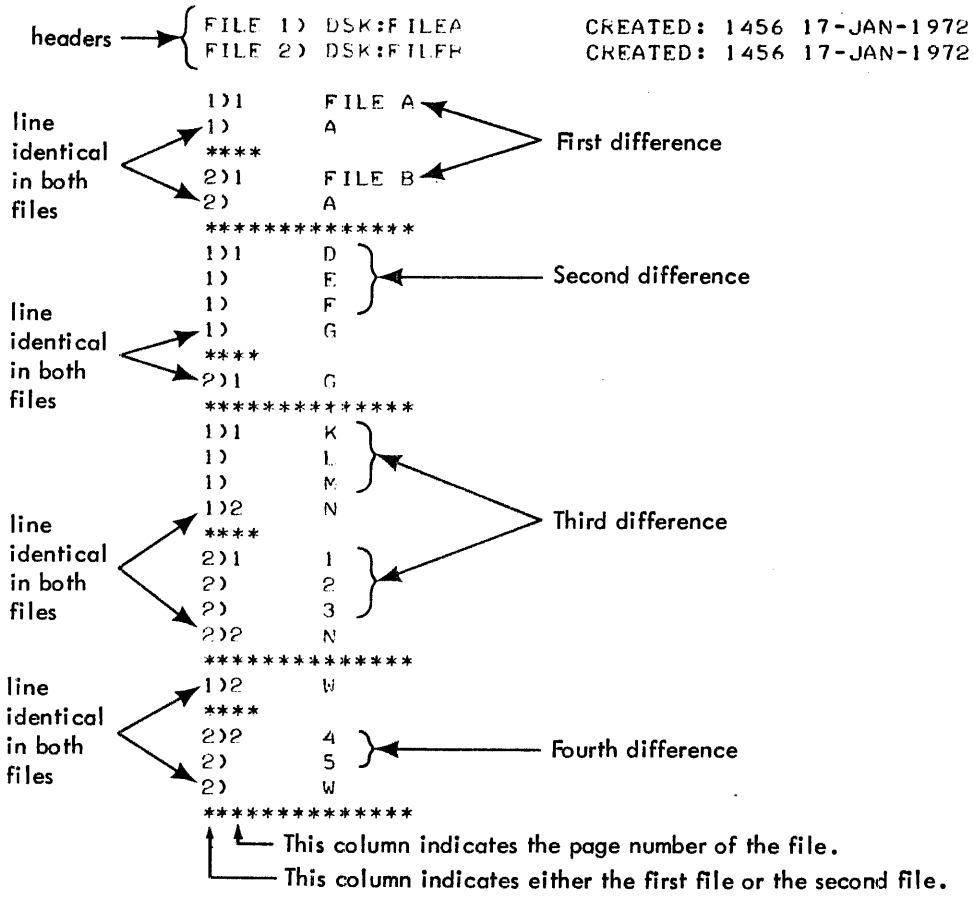
1. The user has the following two ASCII files on disk:



To compare the two files and output the differences on the terminal, the following sequence is used:

.R FILCOM↵	Run the FILCOM program.
*=FILEA,FILEB↵	Compare the two files on disk and output the differences on the terminal. By default, three consecutive identical lines determine a match.

Examples (cont)



Examples (cont)

To compare the two files and output the differences on the line printer, the following commands are used. Note that in this example the number of successive lines that determines a match has been set to 4 with the /4L switch.

```
.R FILCOM
*LPT:/4L=FILEA,FILEB

FILE 1) DSK:FILEA          CREATED: 1456 17-JAN-1972
FILE 2) DSK:FILEB          CREATED: 1456 17-JAN-1972

1)1       FILE A
1)        A
1)        B
1)        C
1)        D
1)        E
1)        F
1)        G
****
2)1       FILE B
2)        A
2)        B
2)        C
2)        G
**************
1)1       K
1)        L
1)        M
1)2       N
****
2)1       1
2)        2
2)        3
2)2       N
**************
1)2       W
****
2)2       4
2)        5
2)        W
**************
```

These lines are listed as being different because the /4L switch specifies that 4 consecutive lines must be found identical in the two files before they are considered as a match.

(continued on next page)

Examples (cont)

To compare the two files so that the second input file is output with vertical bars in the left column next to the lines that differ from the first input file, use the following command sequence.

```
.R FILCOM
*LPT:/U=FILEA,FILEB
```

```
|        FILE B

         A
         B
         C
|        G
         H
         I
         J
|        1
|        2
|        3
         N
         O
         P
         Q
         R
         S
         T
         U
         V
|        4
|        5
         W
         X
         Y
         Z
```

The lines with vertical bars indicate the differences between the two files.

The lines with vertical bars indicate the differences between the two files.

2. To compare two binary files on the disk and output the differences on the terminal, use the following command sequence.

```
.R FILCOM
*TTY:=DSK:DIAL.REL,DIAL2
FILE 1) DSK:DIAL.REL    CREATED: 0000 23-DEC-1971
FILE 2) DSK:DIAL2.REL   CREATED: 0000 12-AUG-1971

000000  000004 000001   000004 000060          000057
000002  000000 054716   000311 372712   000311 326004
000003  000006 000001   017573 510354   017575 510355
000004  000000 000000   017573 513216   017573 513216
```

(continued on next page)

Examples (cont)

3. To compare two high segment files, the command sequence below is used. Note that the locations begin at 400000.

```
.R FILCOM
*TTY:←SYS:TABLE.SHR, TABLE.SHR
FILE 1) SYS:TABLE.SHR    CREATED: 2020 24-JAN-1972
FILE 2) DSK:TABLE.SHR    CREATED: 1829 30-NOV-1971

400000  001611 400010   001630 407157   000021 007147
400003  006675 000000   015024 407670   013651 407670
400004  005600 000070   004700 000113   001100 000163
400005  545741 444562   554143 625700   011602 261262
400010  634000 000000   260740 403516   454740 403516
400011  474000 000000   200000 414036   674000 414036
400012  402000 000156   202000 000720   600000 000676
400013  200040 406354   201000 000472   001040 406726
```

4. To list a binary file, use the following command sequence.

```
.R FILCOM
*TTY:←SYS:DOT.REL
000000  000004 000001
000001  000000 000000
000002  000000 054716
000003  000006 000001
000004  000000 000000
000005  000007 517716
000006  000001 000002
000007  000000 000000
  .
  .
  .
```

Note that the following sequence will not work because of the terminating comma.

```
*TTY:←SYS:DOT.REL,

?COMMAND ERROR
```

Examples (cont)

5. To compare two binary files between locations 150-160 (octal).

```
.R FILCOM
*TTY:/150L/160U=SYS:SYSTAT.SAV,SYS:SYSDPY.SAV
FILE 1) SYS:SYSTAT.SAV  CREATED: 0818 30-NOV-1971
FILE 2) SYS:SYSDPY.SAV  CREATED: 1642 29-NOV-1971

000150  200400 000137   200740 003217   000340 003320
000151  260740 004226   404500 004242   664240 000064
000152  260740 004253   661500 002000   401240 006253
000153  200040 005011   260740 002723   060700 007732
000154  260740 004063   200040 004243   060700 000220
000155  201041 777777   202040 003241   003001 774536
000156  047040 000042   200040 004241   247000 004203
000157  254000 000174   251040 004142   005040 004036
000160  476000 006774   211040 000144   667040 006630
```

6. To compare two .SAV files. Note that the files are expanded before the comparison.

```
.R FILCOM
*TTY:=SYS:TRY1.SAV,SYS:TRY.SAV
FILE 1) SYS:TRY1.SAV    CREATED: 2043 05-JAN-1972
FILE 2) SYS:TRY.SAV     CREATED: 0818 30-NOV-1971

000114  004000 000140   000000 000000   004000 000140
000116  777536 005536   000000 000000   777536 005536
000117  000000 005536   000000 000000          005536
000120  006000 000140   007222 000140   001222 000000
000121  000000 006000   000000 007222          001222
000130  010000 000005   000000 000000   010000 000005
000133  003727 005777   006643 007777   005164 002000
000137  003400 000070   046700 000004   045300 000074
000140  264000 001454   047000 000000   223000 001454
000141  260040 001773   200040 005075   060000 004706
000142  201240 001447   402000 006644   603240 007203
000143  542240 001634   251040 007221   713200 006415
000144  260040 002774   403000 000015   663040 002761
000145  621000 000010   476000 006715   257000 006705
000146  200240 003504   200740 006606   000500 005302
000147  251240 000012   051140 005076   200300 005064
000150  402000 003613   200400 000137   602400 003724
000151  201040 003730   260740 004226   061700 007516
000152  200260 003632   260740 004253   060520 007461
000153  321240 000164   200040 005011   121200 005175
```

Function

The FUDGE2 program is used to update files containing one or more reloctable binary modules and to manipulate the individual modules within these files. Relocatable binary modules are output by MACRO-10, FORTRAN-10, COBOL, ALGOL, and BLISS-10. A module can be a complete program or only a set of subroutines. One reason for collecting a group of relocatable modules into one file is to enable LINK-10 or LOADER to use the file as a library (refer to the LINK-10 or LOADER documentation). Three files are used in the updating process:

1. A master file containing the file to be updated.
2. A transaction file containing the modules to be used when updating.
3. An output file containing the updated file.

All three files can be on the same device if the device is DSK. The two input files can be on the same DECtape.

The desired function of FUDGE2 is specified by a command code at the end of the command string. Only one command code can be specified in each command string. Switches can also be used to position a magnetic tape and to zero a DECtape directory (zeroing a DECtape directory is equivalent to deleting all the files on the tape).

WARNING

For execution to occur, the command string must be terminated with an ALTmode, represented in this manual by a dollar sign ($), instead of the usual carriage return-line feed.

Command Format

```
.R FUDGE2
*output dev:file.ext=master dev:file.ext <modules>, transaction dev:file.ext <modules>
      (command)$
```

output dev: = the device on which the updated file is written. If omitted, DSK is assumed.

master dev: =the device containing the file to be updated. If omitted, the default is DSK. A comma is used to separate the master file and the transaction file.

transaction dev: = the device containing the modules to be used in the updating process. When more than one file is transferred from magnetic tape or paper tape, a colon must follow the device name for each file. For example,

```
MTA: : :  Transfer 3 files
```

If the device is omitted, DSK is assumed.

file. ext =the filename and extension of each file. Filenames must be specified for directory devices, but the extension can be omitted. If the extension is not given, it is assumed to be .REL unless the /L switch appears in the command string. In this case, the output extension .LST is assumed.

Command Format (cont)

file.ext (cont)

Project-programmer numbers appearing after a filename apply to that file only. If the project-programmer number appears before the filename, it applies to all subsequent files until another device is specified.

The protection code of the master file is given to the output file.

The asterisk convention can be used with the input files (refer to Paragraph 1.4.2.4).

<modules>

=Names of modules (on DSK or DTA only) to be used in the updating process. They are grouped within angle brackets in the same order as they appear in the file and are separated by commas. When manipulating all the modules within a file, only the filename need be specified. Module names cannot appear for the output file.

(command)

=Code for the function to be performed. This code can be either preceded by a slash or enclosed in parentheses and must appear at the end of the command string. Each command results in the updated file being output to the output device. The command codes are as follows:

A Append the specified modules in the transaction file(s) to the master file.

C Compress the master file by deleting local symbols. These symbols are included in relocatable binary modules primarily because of their usefulness in debugging procedures. Large libraries of debugged routines, such as LIBOL, frequently have the local symbols deleted in order to save disk space and reduce the amount of I/O required during the loading process.

D Delete the specified modules from the master file.

E Extract the specified files and/or modules from the input files. The entire file is extracted if module names are not specified.

H Type the commands and switches available (help text from device SYS:).

I Insert modules from the specified transaction files into the master file. The modules from the transaction files are inserted immediately before the specified modules in the master file. A comma is used to separate the transaction files.

(continued on next page)

Command Format (cont)

(command) (cont)

L List the names and lengths of all relocatable modules within a file. The length is in one of two forms:

low segment break, high segment break or
program break, absolute break

The length of FORTRAN modules is not output.

The default filename for spooled output is the name of the master file.

R Replace the specified modules in the master file with the specified modules in the transaction file. The number of replacing modules must be the same as the number of modules to replace.

S List all the entry points within a module. These entry points are listed across the page. The default filename for spooled output is the name of the master file.

X Write index blocks into a library file on DECtape or disk. Indexes cannot be written on magnetic tape. Index blocks are used in a direct access library search (refer to the LOADER documentation). This command implies a C command.

The method of numbering the blocks within a file is different on DECtape and disk. This can cause problems with indexed library files that are created on one device and loaded from the other. The index in an indexed library contains the name of each module in the library along with the block number within the library file of the beginning of that module. On disk, the blocks of a file are numbered relative to the beginning of the file; thus, an index references the same blocks properly no matter where the file is placed on the disk. However, on a DECtape, the block numbers are established relative to the beginning of the tape. Therefore, the area of the tape on which the file resides determines the block numbers that will be used in the index. When the LOADER references an indexed library on a device different from the one on which it was created, the block numbers in the index may no longer point to the correct location within the library. This problem can also arise when loading an indexed file that was created at one location on a DECtape and then was transferred to a different location on that tape or to another tape. To transfer indexed files to other devices and then to load them from that device, the index blocks should be deleted before transferring and recreated on the new device. Any FUDGE2 command which generates a new binary file deletes the index blocks and causes a warning message to be output. The /X switch writes the index blocks.

NOTE

An indexed library created on the disk will work properly no matter how many times it is transferred to and from other devices, such as DECtape and magnetic tape, as long as the library is restored to the disk for use by the LOADER or LINK-10.

(continued on next page)

Command Format (cont)

Comments are included on the FUDGE2 command string by preceding the comment with a semicolon. All characters after the semicolon, except for the ALTmode, are ignored until the next line feed, vertical tab, or form feed character is read.

File directories can be manipulated and magnetic tapes positioned by including switches in the command string. These switches can appear anywhere in the command string and are preceded by a slash or enclosed in parentheses. The following switches are available:

/B Backspace a magnetic tape one file.

/K Advance a magnetic tape one file.

/T Skip to the logical end of tape on a magnetic tape.

/W Rewind a magnetic tape.

/Z Clear the directory of the output DECtape.

Characteristics

The R FUDGE2 command:

Places the terminal in user mode.
Runs the FUDGE2 program, thereby destroying the user's core image.

Associated Messages

?CANNOT DO I/O AS REQUESTED

Input (or output) cannot be performed on one of the devices specified for input (output). For example, input may have been requested for a device that can only do output.

?COMMAND SWITCH REQUIRED

The given command string requires a FUDGE2 command code.

?DEVICE ERROR ON OUTPUT DEVICE

A write error has occurred on the output file.

?DIRECTORY FULL ON OUTPUT DEVICE

There is no room in the file directory on the output device to add the updated file (non-disk devices only).

Associated Messages (Cont)

?ENTER FAILURE n

The output filename is null; n is the error code for an illegal filename (non-disk devices only).

?ENTER FAILURE

The ENTER to write the disk file failed. This message is followed by a line explaining the reason for failure.

?ENTRY BLOCK TOO LARGE, PROGRAM name

The entry block of the named program is too large for the FUDGE2 entry table, which allows for 100 entry names. FUDGE2 can be reassembled with a larger table.

?FUDGE2 SYNTAX ERROR

An illegal command string was entered; for example, the left arrow was omitted or a program name was specified for the output file.

?ILLEGAL BLOCK TYPE dev:file.ext

The block type used is not in the range 0-77.

?ILLEGAL DATA MODE FOR dev

The data mode specified for a device in the user's program is illegal, such as dump mode for the terminal.

?(0)ILLEGAL FILENAME

A filename of zero was specified.

?INPUT ERROR ON DEVICE dev: STATUS (nnnnnn)

A data or device error occurred on input.

?x IS AN ILLEGAL { CHARACTER / SWITCH }

An illegal character or switch was encountered in the command string.

?LOOKUP FAILURE

The LOOKUP to read the disk file failed. This message is followed by a line explaining the reason for failure.

Associated Messages (Cont)

?dev:file.ext <> NO PROGRAM NAME SPECIFIED

The switch (/D or /R) used in the command string requires that a program name be given.

?dev NOT AVAILABLE

The specified device does not exist or is assigned to another user.

?NOT ENOUGH ARGUMENTS

An insufficient number of files of one type has been specified.

?dev file.ext program NOT FOUND

The file or the program was not found on the device or in the file specified. If a program name is printed, this message may indicate that the program names in the command string appear in a sequence different from their sequence within the file. Therefore, the program may actually exist but was missed because of the incorrect sequence in the command string.

?PROGRAM ERROR WHILE RESETTING MASTER DEVICE

FUDGE2 cannot find the master device or cannot find the program on the master device.

?TOO MANY FILENAMES OR PROGRAM NAMES

More than 40 program names or filenames were specified in the command string. The user should separate the job into several segments.

?TRANSMISSION ERROR ON INPUT DEVICE dev

A transmission error has occurred while reading data from the specified device.

?UNEQUAL NUMBER OF MASTER AND TRANSACTION PROGRAMS

On a replace request, the number of master programs (or files) does not equal the number of transaction programs (or files).

WARNING NO INDEX ON OUTPUT FILE-CONTINUING

The user has changed the structure of the indexed library file when deleting, appending, or inserting, thereby invalidating the index. The index has been removed from the new file. Reindexing is required.

Examples

```
.R FUDGE2
*LPT:=DTA1:LIB40(L)$
```

List all relocatable modules from the file LIB40 .REL, located on DTA1, on the line printer.

```
*DSK:LIB4BB=DTA2:LIB4AA <EXP.3,EXP.3C>,
DTA1:F4<EXP.3A,EXP.3B>(R)$
```

Replace modules EXP.3 and EXP.3C located in file LIB4AA.REL on DTA2, with modules EXP.3A and EXP.3B in File F4.REL on DTA1; write out the new LIB4AA file on disk and call it LIB4BB.REL.

```
*DTA1:NFILE=DSK:MFILE<M1,M2,M3,M4>,
DTA3:TFILEA<TA1,TA2>,
DTA4:TFILEB<TB1,TB2>/I$
```

Insert into MFILE the modules TA1 and TA2 from TFILEA, and TB1 and TB2 from TFILEB. Create NFILE with the following order:

TA1, M1, TA2, M2, TB1, M3, TB2, M4

Insertion is on a one-to-one basis. If there are more modules to be inserted than specified modules before which they are to be inserted, the extra files are ignored.

```
*DTA1:NFILE=DSK:MFILE<M1,M2,M3,M4>,
DTA3:TFILEA,
DTA4:TFILEB/I$
```

However, in this example (where TFILEA.REL and TFILEB.REL contain the modules TA1 and TA2 and TB1 and TB2, respectively) create an NFILE.REL with the following order:

TA1,TA2,M1,TB1,TB2,M2,M3,M4

```
*DTA2:TESTA=MTA1:(WK),MTA2: :(ZA)$
```

Clear the directory of DTA2; rewind MTA1 and advance the tape one file; append the first two program files from MTA2 to the second file on MTA1 and write out the resultant file on DTA2, calling it TESTA.REL.

```
*OUTPUT=LIBARY,DTA1:LIBARY<FILEY,FILEZ>/A$
```

Append the modules FILEY and FILEZ contained in the file LIBARY.REL on DTA1 to the end of the file LIBARY.REL on disk. Write the new file on disk and call it OUTPUT.REL.

```
*NEWFIL=OLDFIL<TEST,SUBTRC,MULTI>,BASFIL<PROG,
ROUTIN,ANSWER>,SUBFIL<MATH>(E)$
```

Extract the specified modules from the files OLDFIL, BASFIL, and SUBFIL and create a new output file called NEWFIL. The order of the modules in NEWFIL is as follows: TEST, SUBTRC, MULTI, PROG, ROUTIN, ANSWER, MATH.

(continued on next page)

Examples (cont)

```
*NEWF40=DTA2:OLDF40<SUBTLE,DATFIL,ROUTNE>/D$
```

Delete the modules SUBTLE, DATFIL, and ROUTNE from the file OLDF40.REL on DTA2 and create a new output file NEWF40.REL on disk containing the remainder of file OLDF40.

```
*NOIDX.REL=IDX.REL(A)$
```

Delete index blocks from the file IDX.REL and write the remainder of the file on the output file NOIDX.REL. The Append command (A) generates a new binary file and therefore removes the index blocks.

```
*↑C
```

Return to the monitor.

Function

The GLOB program reads multiple binary program files and produces an alphabetical cross-referenced list of all the global symbols (symbols accessible to other programs) encountered. This program also searches files in library search mode, checking for globals, if the program file was loaded by the LOADER in library search mode (refer to the LOADER documentation).

The GLOB program has two phases of operation; the first phase is to scan the input files and build an internal symbol table, and the second, to produce output based on the symbol table. Because of these phases, the user can input commands to GLOB in one of two ways. The first way is to specify one command string containing both the output and input specifications. (This is the command string format most system programs accept.) The second is to separate the command string into a series of input commands and output commands.

Command Formats

1. R GLOB

   outdev:file.ext [directory] = input dev:file.ext [directory], file.ext,...,dev:file.ext [directory] (\$)

2. R GLOB

   followed by one or more input commands in the form

   dev:file.ext [directory],file.ext [directory],...,dev:file.ext [directory],...

   and then one or more output commands in the form

   outdev:file.ext [directory] =(\$)

When the user separates his input to GLOB into input commands and output commands (Command Format #2), the input commands contain only input specifications and the output commands, only output specifications. Each output command causes a listing to be generated; any number of listings can be printed from the symbol table generated from the current input files as long as no input commands occur after the first output command. When an input command is encountered after output has been generated, the current symbol table is destroyed and a new one begun.

Defaults

1. If the device is omitted, it is assumed to be DSK. However, if the entire output specification is omitted, the output device is TTY.

(continued on next page)

Command Format (cont)

Defaults (cont)

2. If the output filename is omitted, it is the name of the last input file on the line (Command Format #1) or is GLOB if the line contains only output commands (Command Format #2). The input filenames are required.
3. If the output extension is omitted, .GLB is used. If the input extension is omitted, it is assumed to be .REL unless the null extension is explicitly specified by a dot following the filename.
4. If the project-programmer number [proj,prog] is omitted, the user's default directory is used.
5. An ALTmode terminates the command input and signals GLOB to output the cross-referenced listing. In other words, a listing is not output until GLOB encounters an ALTmode. The ALTmode appears at the end of the command string shown in Command Format #1 or at the end of each output command shown in Command Format #2.

Switches

Switches control the types of global listings to be output. Each switch can be preceded by a slash, or several switches can be enclosed in parentheses. Only the most recently specified switch (except for L, M, P, Q, and X, which are always in effect) is in effect at any given time. If no switches are specified, all global symbols are output. The following switches are available.

/A	Output all global symbols. This is the default if no switches are specified.
/E	List only erroneous (multiple defined or undefined) symbols.
/F	List nonrelocatable (fixed) symbols only.
/H	List the switches available (help text) from SYS:GLOB.HLP.
/L	Scan programs only if they contain globals previously defined and not yet satisfied (library search mode).
/M	Turn off library search mode scanning resulting from a /L switch.
/N	List only symbols which are never referenced.
/P	List all routines that define a symbol to have the same value. The routine that defines the symbol first is listed followed by a plus (+) sign. Subsequent routines that define the symbol are listed preceded by a plus sign.
/Q	Suppress the listing of subsequent definers that result from the /P switch.

(continued on next page)

Command Format (cont)

Switches (cont)

/R List only relocatable symbols.

/S List symbols with non-conflicting values that are defined in more than one program.

/X Do not print listing header when output device is not the terminal, and include listing header when it is the terminal. Without this switch, the header is printed on all devices except the terminal. The listing header is in the following format:

FLAGS SYMBOL OCTAL VALUE DEFINED IN REFERENCED IN

Symbols listed are in alphabetical order according to their ASCII code values. The octal value is followed by a prime (') if the symbol is relocatable. The value is then relative to the beginning of the program in which the symbol is defined. Flags preceding the symbol are shown below.

M Multiply defined symbol (all values are shown).

N Never referred to (i.e., was not declared external in any of the binary programs).

S Multiply specified symbol (i.e., defined in more than one program but with non-conflicting values). The name of the first program in which the symbol was encountered is followed by a plus sign.

U Undefined symbol.

Characteristics

The R GLOB command:

Places the terminal in user mode.
Runs the GLOB program, thereby destroying the user's core image.

Associated Messages

?COMMAND SYNTAX ERROR
TYPE/H FOR HELP

An illegal command string was entered.

?DESTINATION DEVICE ERROR

An I/O error occurred on the output device.

Associated Messages (cont)

?ENTER ERROR n
?DIRECTORY FULL

No additional files can be added to the directory of the output device; n is the disk error code.

?ILLEGAL SWITCH

A non-recognizable switch was used in the command string.

?LOOKUP ERROR n
?file.ext FILE NOT FOUND

The named file cannot be found in the directory on the specified device.

?dev NOT AVAILABLE

The requested device does not exist or is assigned to another job.

?TABLE OVERFLOW - CORE UUO FAILED TRYING TO EXPAND TO xxx

The GLOB program requested additional core from the monitor, but none was available.

Examples

.R GLOB↵	Run the GLOB program.
*LPT:=MAIN,DTA2:SUB40,SUB50 (S)	All global symbols in the programs MAIN (on DSK), SUB40, and SUB50 (on DTA2) are listed on the line printer. Along with the symbol is listed its value, the program in which it is defined, all programs in which it is referenced, and any error flags.
*DTA4:BATCH.REL,DATA,DTA6:NUMBER.REL,CLASS↵	
*DSK:MATH.REL,LIBARY.↵	The programs to be scanned are BATCH.REL, DATA.REL on DTA4; NUMBER.REL, CLASS.REL on DTA6; and MATH.REL, LIBARY.null on DSK.
*LPT:=/F (S)	List only nonrelocatable symbols on the line printer.
*DSK:SYMBOL=/R (S)	List only relocatable symbols in the file named SYMBOL in the user's default directory.
*TTY:=/E (S) U EXTSYM SUBRTE	Print all erroneous symbols on the terminal. EXTSYM is an undefined symbol appearing in the program SUBRTE.
*↑C	Return to monitor mode.