K 7

```
RRRRRRRRRRRR     UUU         UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF  FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU         UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF  FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU         UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF  FFFFFFFFFFFFFFF
RRR         RRR  UUU         UUU  NNN         NNN  000         000  FFF              FFF
RRR         RRR  UUU         UUU  NNN         NNN  000         000  FFF              FFF
RRR         RRR  UUU         UUU  NNN         NNN  000         000  FFF              FFF
RRR         RRR  UUU         UUU  NNNNNN      NNN  000         000  FFF              FFF
RRR         RRR  UUU         UUU  NNNNNN      NNN  000         000  FFF              FFF
RRR         RRR  UUU         UUU  NNNNNN      NNN  000         000  FFF              FFF
RRRRRRRRRRRR     UUU         UUU  NNN   NNN   NNN  000         000  FFFFFFFFFFFF     FFFFFFFFFFFF
RRRRRRRRRRRR     UUU         UUU  NNN   NNN   NNN  000         000  FFFFFFFFFFFF     FFFFFFFFFFFF
RRRRRRRRRRRR     UUU         UUU  NNN   NNN   NNN  000         000  FFFFFFFFFFFF     FFFFFFFFFFFF
RRR   RRR        UUU         UUU  NNN      NNNNNN  000         000  FFF              FFF
RRR   RRR        UUU         UUU  NNN      NNNNNN  000         000  FFF              FFF
RRR   RRR        UUU         UUU  NNN      NNNNNN  000         000  FFF              FFF
RRR      RRR     UUU         UUU  NNN         NNN  000         000  FFF              FFF
RRR      RRR     UUU         UUU  NNN         NNN  000         000  FFF              FFF
RRR      RRR     UUU         UUU  NNN         NNN  000         000  FFF              FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN         NNN     000000000     FFF              FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN         NNN     000000000     FFF              FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN         NNN     000000000     FFF              FFF
```

```
_$2

Sym
---
NDX
NDX
NUM
NUM
OPE
OUT
PAC
PAC
PAC
PAC
PAC
PAC
PAC
PAD
PAG
PAG
PAG
PAG
PAG
PAG
PAG
PER
PUT
RCO
RIN

RLI
RNO
RNO
RTY
SAV
STR
STR
STR
STR
STR
STR

STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
```

```
**FILE**ID**RUNOFF                                J 14
```

```
RRRRRRRR    UU      UU  NN      NN    OOOOOO    FFFFFFFFFF  FFFFFFFFFF
RRRRRRRR    UU      UU  NN      NN    OOOOOO    FFFFFFFFFF  FFFFFFFFFF
RR      RR  UU      UU  NN      NN  OO      OO  FF          FF
RR      RR  UU      UU  NN      NN  OO      OO  FF          FF
RR      RR  UU      UU  NNNN    NN  OO      OO  FF          FF
RR      RR  UU      UU  NNNN    NN  OO      OO  FF          FF
RRRRRRRR    UU      UU  NN  NN  NN  OO      OO  FFFFFFFF    FFFFFFFF
RRRRRRRR    UU      UU  NN  NN  NN  OO      OO  FFFFFFFF    FFFFFFFF
RR  RR      UU      UU  NN    NNNN  OO      OO  FF          FF
RR  RR      UU      UU  NN    NNNN  OO      OO  FF          FF
RR    RR    UU      UU  NN      NN  OO      OO  FF          FF              ....
RR    RR    UU      UU  NN      NN  OO      OO  FF          FF              ....
RR      RR  UUUUUUUUUU  NN      NN    OOOOOO    FF          FF              ....
RR      RR  UUUUUUUUUU  NN      NN    OOOOOO    FF          FF              ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

RU
VO

```
  0001  0 %TITLE 'Overall DSR process control'
  0002  0 MODULE runoff ( IDENT = 'V04-000'
C 0003  0             %BLISS32[, ADDRESSING_MODE (EXTERNAL    = general %(,
  0004  0                                          NONEXTERNAL = long_relative)%)]
  0005  0             ) =
  0006  1 BEGIN                                               !Start of module
  0007  1
  0008  1 !
  0009  1 !****************************************************************************
  0010  1 !*                                                                          *
  0011  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
  0012  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
  0013  1 !*  ALL RIGHTS RESERVED.                                                    *
  0014  1 !*                                                                          *
  0015  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
  0016  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
  0017  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
  0018  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
  0019  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
  0020  1 !*  TRANSFERRED.                                                            *
  0021  1 !*                                                                          *
  0022  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
  0023  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
  0024  1 !*  CORPORATION.                                                            *
  0025  1 !*                                                                          *
  0026  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
  0027  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
  0028  1 !*                                                                          *
  0029  1 !*                                                                          *
  0030  1 !****************************************************************************
  0031  1 !
  0032  1 !++
  0033  1 ! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
  0034  1 !
  0035  1 ! ABSTRACT:     Main program.
  0036  1 !
  0037  1 ! ENVIRONMENT:  Transportable
  0038  1 !
  0039  1 ! AUTHOR:       R.W.Friday      CREATION DATE:  April, 1978
  0040  1 !
```

```
1 !
1 ! MODIFIED BY:
1 !
1 !       040     REM00040        Ray Marshall    25-Mar-1984
1 !               Fixed the call to LIB$FIND_FILE.  The new user flags parameter
1 !                 must be passed by reference, not value.
1 !
1 !       039     REM00039        Ray Marshall    16-Feb -> 15-Mar-1984
1 !               Added routine INPUT_FILE_SCAN to make appropriate calls to
1 !                 LIB$FIND_FILE.  This is code to support directory search
1 !                 string logic implemented in VMS V4, and as such, is
1 !                 conditionalized to only compile for VMS and will only
1 !                 excute on a V4 or later system.  It will only be compiled
1 !                 into BLISS32 implementations.
1 !
1 !       038     KFA00038        Ken Alden       12-Aug-1983
1 !               Fixed a /auto bug that forces a second run if any crefs
1 !                 were unresolved (usu. a forward reference).
1 !
1 !       037     REM00037        Ray Marshall    9-Aug-1983
1 !               Modified the init-file logic to support DSRPLUS.INIT and
1 !                 RUNOFF.INIT in the TOPS-20 implementation of DSRPLUS
1 !                 (replacing the logicals used in the VMS implementation).
1 !
1 !       036     REM00036        Ray Marshall    22-Jul-1983
1 !               Defined TTYSET and ENAPSI as NOVALUE to match the way they
1 !                 are coded.
1 !               Moved GCA_PASS_COUNT out of being BLISS32 specific because
1 !                 it is looked at in the routine OUTCREF which is part of all
1 !                 implementations of DSRPLUS.
1 !
1 !       035     KFA00035        Ken Alden       15-Jul-1983
1 !               Tweaked the /auto logic to prevent a premature MEM file.
1 !
1 !       034     KFA00034        Ken Alden       28-Jun-1983
1 !               Added formal parameter to the SETQUICK call.
1 !
1 !       033     KFA00033        Ken Alden       27-Jun-1983
1 !               Added more logic to the setquick check using the cref_errcnt
1 !                 as an additional check item.
1 !
1 !       032     KFA00032        Ken Alden       24-Jun-1983
1 !               This edit change did exactly what 31 was supposed to do.
1 !                 The big problem was the gca_black_box was getting turned
1 !                 off after the first run, from RINIT.  Now, gca_black_box
1 !                 is not check, the rno_cmd[automatic] is.
1 !
1 !       031     KAD00031        Keith Dawson    7-Jun-1983
1 !               Minor tweak to the foregoing: make sure no .MEM is gen-
1 !                 erated (QUICK is set) on reruns caused because a cross-
1 !                 reference changed.
1 !
1 !       030     KAD00030        Keith Dawson    27-May-1983
1 !               For DSRPLUS/AUTO, check number of iterations for Cref oscil-
1 !                 lation; quit after the 4th pass.
1 !
1 !       029     KAD00029        Keith Dawson    17-May-1983
```

```
1 !                     For FLIP, add TCXINI record to initialization records
1 !                       written to the .BFL file.
1 !
1 !             028     REM00028        Ray Marshall    10-May-1983
1 !                     RNFXFL has been removed, so the reference herein used to check
1 !                       for parallelism of MSGTXT.REQ (via RNFERM.BLI) with
1 !                       DSRMSG.MSG (or DPLMSG.MSG for DSRPLUS) has been changed to
1 !                       refer to RNFWFR.
1 !
1 !             027     KAD00027        Keith Dawson    3-May-1983
1 !                     Fix bug: if /AUTO, any errors in Pass 1 would stop processing.
1 !
1 !             026     KAD00026        Keith Dawson    19-Apr-1983
1 !                     Fixed bug whereby the results of /DEVICE were not getting
1 !                       picked up in DSR.
1 !
1 !             025     KAD00025        Keith Dawson    5-Apr-1983
1 !                     Full support for /CROSS and /AUTO.
1 !
1 !             024     KAD00024        Keith Dawson    23-March-1983
1 !                     Changed GCA_FLIP bit to (.gca_op_dev EQL op_dev_flip).
1 !
1 !             023     REM00023        Ray Marshall    07-Mar-1983
1 !                     Global edit of all modules. Updated module names, idents,
1 !                     copyright dates. Changed require files to BLISS library.
1 !--
```

```
     1 !
     1 ! TABLE OF CONTENTS:
     1 !
     1
     1 REQUIRE 'REQ:RNODEF';                         ! RUNOFF variant definitions
     1
     1 FORWARD ROUTINE
     1     runoff,
     1     grab_resultant,
U    1 %IF DSRPLUS %THEN
U    1     init_files : NOVALUE,
     1 %FI
     1 %IF %BLISS(BLISS32) %THEN
     1     input_file_scan,
     1 %FI
     1     input_loop;
     1
     1 !
     1 ! INCLUDE FILES:
     1 !
     1
     1 LIBRARY 'SYS$LIBRARY:STARLET';
     1
     1 undeclare %quote $DESCRIPTOR;
     1 !undeclare %quote RMS$_FNF;
     1 !undeclare %quote RMS$_NORMAL;
     1
     1 LIBRARY 'NXPORT:XPORT';                       ! XPORT Library
     1
U    1 %IF DSRPLUS %THEN
U    1 LIBRARY 'REQ:DPLLIB';                        ! DSRPLUS BLISS Library
     1 %ELSE
     1 LIBRARY 'REQ:DSRLIB';                        ! DSR BLISS Library
     1 %FI
     1
     1 !
     1 ! MACROS:
     1 !
     1 ! This macro returns "1" if RUNOFF is running on TOPS-10 or TOPS-20.
     1 ! If we're running on VAX/VMS, the specified return code is used,
     1 ! with bit 28 set to inhibit generation of an error message.
     1 ! On other systems the specified code is returned without modification.
     1 !
     1 MACRO
M    1     creturn (return_code) =
M    1         %IF %BLISS (BLISS36) %THEN
M    1             RETURN 1
M    1         %ELSE
M    1             RETURN return_code
M    1                 + %IF %BLISS(BLISS32) %THEN 1^28 ! Set inhibit message bit
M    1                   %ELSE 0
M    1                   %FI
M    1         %FI
     1                                 %,      ! End of macro CRETURN
     1
M    1     close_i_o_and_return (delete, return_value) =
M    1         BEGIN
```

```
M  1            clh (clh_close_input);
M  1            ! Close output file unless quick processing (no output file opened).
M  1            IF  NOT .rno_cmd [rno$v_quick]  THEN
M  1                ! Delete output file if user requested.
M  1                BEGIN
M  1                IF delete
M  1                THEN  clh (clh_close_del_out)
M  1                ELSE  clh (clh_close_out);
M  1                END;
M  1            creturn (return_value);
M  1            END
   1                                           %;       !  End of macro CLOSE_I_O_AND_RETURN
   1  !
   1  ! OWN STORAGE:
   1  !
   1  OWN
U  1  %IF DSRPLUS %THEN
U  1       init_file_error,                ! Error flag for opening DSR[PLUS]$INIT
   1  %FI
   1       file_error,                     ! Error flag for main file processing
   1       tempob : $XPO_IOB (),           ! IOB for output file
   1       tempib : $XPO_IOB ();           ! IOB for primary input file
   1
   1  !
   1  ! EXTERNAL REFERENCES:
   1  !
   1  EXTERNAL
   1       rneiob : REF $XPO_IOB (),
   1       rnaiob : REF $XPO_IOB (),
   1       rnoiob : REF $XPO_IOB (),
   1       tteiob : $XPO_IOB (),
   1       ffname : $STR_DESCRIPTOR (CLASS = dynamic), ! Failing file's name
   1       semcod,                                     ! Secondary Error Message CODe
   1       gca  : gca_definition,
   1       ira  : fixed_string,
   1       irac : irac_definition,
   1       fs01 : fixed_string,
   1       lffctx,                         ! LIB$FIND_FILE context cell
   1       ipftyp,                         ! InPut File TYPe index
   1       ipftop : VECTOR,                ! InPut File Type OPtions list
   1       khar;
   1
   1
   1  EXTERNAL ROUTINE
   1       clh,        doopts,        erm,            erme,
   1       erms,       pus,           rinit,          rterm
   1
   1  %IF %BLISS (BLISS32) %THEN  ,
   1
U  1  %IF DSRPLUS %THEN
U  1       setquick,
   1  %FI
   1       LIB$FIND_FILE                   ! NEEDED to resolve search strings on VMS!
   1  %FI
   1
U  1  %IF (%BLISS(BLISS36) AND DSRPLUS)  %THEN   ,
U  1       enapsi : NOVALUE,               ! Enable CTRL/C PSI interrupt.
```

```
U  1      ttyset : NOVALUE              ! Set (save) terminal characteristics.
   1 %FI                       ;
   1
   1 EXTERNAL LITERAL
   1      rnfcoi,       ! F - Can't open input file "%S"
   1      rnfcoo,       ! F - Can't open output file "%S"
   1      rnffab,       ! W - File aborted
   1      rnfrtl,       ! W - Input record too long: truncated "%S"
   1      rnfure        ! W - Unrecoverable error processing record %C on page %I of input file "%F"
   1
   1 %IF %BLISS(BLISS32) %THEN
   1              ! Special error messages used to detect and signal bad error message
   1              ! pointers:
   1      rnfwfr,       ! Last error defined by the message utility.
   1      last_rnf,     ! Last error defined by RNFERM and MSGTXT.REQ
   1      rnfaaa,       ! Message used to signal the inequality of the previous two
   1
U  1 %IF dsrplus %THEN
U  1      rnfoft,       ! File type ".RNO" not found.  Processing file "%F"
   1 %FI
   1 !     RMS$_FNF,   ! VMS RMS error code:  File Not Found
   1 !     RMS$_NORMAL,! VMS RMS return code:  Normal return
   1 %FI
   1      ipftct;       ! Literal defining the lengths of IPFTOP and OPFTOP.
   1
```

```
  1 GLOBAL ROUTINE runoff (rno_cmd) =
  1
  1 !++
  1 ! FUNCTIONAL DESCRIPTION:
  1 !
  1 !       This is the main program of RUNOFF.  The basic processing involves
  1 !       calling CLH to open and close files, and read records that get passed
  1 !       to PUS.  RINIT and RTERM are only marginally related to the formatting
  1 !       processing, being used for initialization and termination.
  1 !
  1 ! FORMAL PARAMETERS:     None
  1 !
  1 ! IMPLICIT INPUTS:       None
  1 !
  1 ! IMPLICIT OUTPUTS:      None
  1 !
  1 ! ROUTINE VALUE:
  1 ! COMPLETION CODES:      None
  1 !
  1 ! SIDE EFFECTS:          None
  1 !--
  1
  2     BEGIN                                        !Start routine RUNOFF
  2
  2     MAP
  2         rno_cmd : REF $rno_cmd;
  2     LOCAL
  2 %IF %BLISS(BLISS32) %THEN
  2         status,                 ! Return value from INPUT_FILE_SCAN routine
  2         input_file_name :       ! Receiving string descriptor
  2                 $STR_DESCRIPTOR (CLASS = dynamic),
  2 %FI
  2         clh_result,             ! Result of CLH operation
  2         keep_reading,           ! Controls input-reading loop
  2         hold_cref_errcnt,       ! Remembers gca_cref_errcnt over RINIT.
  2         hold_cref_changed;      ! Remembers gca_cref_changed over RINIT
  2
U 2 %IF DSRPLUS %THEN
U 2         !+
U 2         ! Initialize automatic variables
U 2         !-
U 2         gca_pass_count = 1;                  ! Current pass over input file
U 2 %IF %BLISS (BLISS32) %THEN
U 2         gca_rerun_count = 0;                 ! Number of times input file must be reprocessed
U 2         gca_cref_errcnt = 0;                 ! No cross-references have been unresolved yet.
U 2         gca_cref_changed = false;            ! No cross-references have changed.
U 2
U 2     WHILE true DO
U 2         BEGIN                                ! Start of automatic-processing loop
U 2             !
U 2             ! Reprocess the input file until exit condition is met
U 2             !
U 2         hold_cref_errcnt = .gca_cref_errcnt;
U 2         hold_cref_changed = .gca_cref_changed;
  2 %FI  %FI
  2     rneiob = tempib;
  2     rnaiob = tempob;
```

```


    $XPO_IOB_INIT (IOB = .rnaiob, FILE_SPEC = rno_cmd [rno$t_output]);

    rinit ();                            ! Initialize most everything.

%IF %BLISS(BLISS32) %THEN
    !+
    ! Now we must see if the VMS message utility using DSRMSG.MSG as input has
    ! defined the same number of messages as RNFERM.BLI has put into the message
    ! vector using MSGTXT.REQ as input.  This is done by verifying that a global
    ! defined in RNFERM.BLI (LAST_RNF) is equal to the last message code defined
    ! by the message utility (RNFWFR).  If not, signal the FATAL error and exit!
    !-
    IF rnfwfr NEQ last_rnf
    THEN
        BEGIN
        erm (rnfaaa, 0, 0);
        creturn (4)
        END;
%FI

%IF %BLISS(BLISS32) %THEN
    $STR_DESC_INIT( DESCRIPTOR = input_file_name
                   ,CLASS = DYNAMIC);
                                         ! Locate an input file
    IF (status = input_file_scan (.rno_cmd, input_file_name)) EQL 4
        THEN creturn(.status)
    ELSE
        $XPO_IOB_INIT (IOB = .rneiob, FILE_SPEC = input_file_name);
%ELSE
    $XPO_IOB_INIT (IOB = .rneiob, FILE_SPEC = rno_cmd [rno$t_input]);
%FI
    !
    ! Now proceed with normal processing.
    !

    clh_result = clh (clh_open_input);  ! Try to open input file.

    IF  .clh_result NEQ clh_normal
    THEN
        BEGIN                                    ! Start can't open input
        LOCAL
            spec_length,
            spec_ptr;

        spec_length = .ffname [str$h_length];
        spec_ptr    = .ffname [str$a_pointer];
        !
        erme (rnfcoi, .spec_ptr, .spec_length, .semcod);

        creturn (4)
        END                                      ! End can't open input
    ELSE
    ! Did user specify quick processing? If so, don't open an output file.
        IF  .rno_cmd [rno$v_quick]
        THEN
```

```
  2             clh_result = clh_normal
  2         ELSE
  3             BEGIN
  3             !
  3             ! Pick up command-line information about /DEVICE now,
  3             ! so that output file defaults can be set up properly.
  3             !
  3             gca_op_dev   = .rno_cmd [rno$v_4_out_format]; ! Pick up device type.
  3             gca_s_output = .rno_cmd [rno$v_s_output];     ! See if user said /OUTPUT=file.
  3             clh_result   = clh (clh_open_out);  ! Open output file.
  3
U 3 %IF (%BLISS(BLISS36) AND DSRPLUS)  %THEN
U 3             !+
U 3             ! The following call modifies the Control Character Output
U 3             ! Control (CCOC) words to allow escape sequences to be sent to
U 3             ! the terminal without translation, and modifies the JFN mode
U 3             ! word to prevent TOPS-20 from wrapping output lines at the
U 3             ! terminal width.
U 3             !-
U 3             IF  (.gca_op_dev EQL op_dev_vt100)  ! User said /DEC=VT100, and did
U 3                 AND NOT .gca_s_output           ! not say /OUTPUT=name.
U 3             THEN
U 3                 ttyset ();                      ! Set terminal characteristics.
U 3             !
U 3             ! Enable the PSI CTRL/C interrupt.
U 3             !
U 3             enapsi ();
  3 %FI
  2             END;
  2
  2     IF  .clh_result NEQ clh_normal
  2     THEN
  3         BEGIN                                       ! Start can't open output
  3         LOCAL
  3             spec_length,
  3             spec_ptr;
  3
  3         spec_length = .ffname [STR$H_LENGTH];
  3         spec_ptr    = .ffname [STR$A_POINTER];
  3         !
  3         ! Can't open output file.
  3         !
  3         erme (rnfcoo, .spec_ptr, .spec_length, .semcod);
  3         clh (clh_close_input);
  3         creturn (4)
  3         END                                         ! End can't open output
  2     ELSE
  3         BEGIN                                       ! Start normal file processing
U 3 %IF FLIP  %THEN
U 3         IF  (.gca_op_dev EQL op_dev_flip)       ! User said /DEVICE=FLIP
U 3         THEN
U 3             BEGIN
U 3             LOCAL
U 3                 init_record : $flip_init,
U 3                 tcxini_record : $flip_tcxini;
U 3
U 3         ! Write FLIP initialization record.
```

```
U 3         !
U 3             init_record [init_code]     = flip$k_init;
U 3             init_record [init_contents] = .rno_cmd [rno$v_intermediate];
U 3             init_record [init_level]    = flip$k_version;
U 3             rno_cmd [rno$v_intermediate]    = 1;
U 3             $XPO_PUT (  IOB    = .rnoiob,
U 3                         STRING = (flip$k_init_size,
U 3                         CH$PTR (init_record)));
U 3
U 3         ! Write FLIP index-information initialization record.
U 3         !
U 3             tcxini_record[tcxini_code] = flip$k_tcxini;
U 3             tcxini_record[tcxini_header] = index_format;
U 3             $XPO_PUT (  IOB     = .rnoiob,
U 3                         STRING  = (flip$k_tcxini_size,
U 3                         CH$PTR (tcxini_record)));
U 3             END;
  3 %FI
  3
U 3 %IF (%BLISS (BLISS32) AND DSRPLUS) %THEN
U 3             ! This logic makes sure we don't create a .MEM file for an
U 3             ! intermediate run of /AUTO -- if the reason we need one more
U 3             ! run is because a cross-reference changed.
U 3             !
U 3             IF  .rno_cmd [rno$v_automatic]! gca_black_box  ! Are we doing /AUTO?
U 3                 AND
U 3                 ((.hold_cref_changed EQL true) OR
U 3                 ((.hold_cref_errcnt GTR 0) AND (.gca_pass_count EQL 2)))  ! One changed or was unresolved on
U 3                 AND .gca_pass_count NEQ 4 ! Last chance, oscillating cref's.
U 3             THEN
U 3                 setquick (true);
U 3         !
U 3         ! The user may have had some forward crefs.
U 3         ! We give him the benefit of the doubt and run plus again.
U 3         !
U 3         IF (.hold_cref_errcnt GTR 0) AND (.gca_pass_count EQL 2)
U 3         THEN
U 3             gca_rerun_count = .gca_rerun_count + 1;
  3 %FI
  3         IF  NOT doopts (.rno_cmd)
  3         THEN
  3             !
  3             ! Close and delete output file; return 4.
  4             close_i_o_and_return (true, 4)
  3         ELSE
  4             BEGIN
  4             !+
  4             ! If the output file is the terminal, don't redirect error messages.  Otherwise, direct messages
  4             ! specified on the /MESSAGES switch.  The effect is that the user can determine where error mess
  4             ! go only if the output file is not the terminal; i.e., you cannot suppress error messages total
  4             ! but you can prevent them from going into the output file.
  4             !-
  5             IF  NOT (.rnoiob [iob$v_terminal] AND
  5                 .tteiob [iob$v_terminal])
  4             THEN
  4                 gca_err_dir =.gca_cmd_msg;
U 4 %IF DSRPLUS %THEN
```

```
U 4                    ! Attempt to open initialization files DSR$INIT and/or
U 4                    ! DSRPLUS$INIT.
U 4                    !
U 4                    init_files (.rno_cmd);
  4 %FI
  3                    END;
  3
  3 !+
  3 ! Here is RUNOFF's main processing loop.
  3 !-
  3            keep_reading = true;
  3            file_error = false;
  3
  3            WHILE .keep_reading DO
  3                keep_reading = input_loop (file_error);
  3            !
  3            ! Check status of file errors, and finish up accordingly.
  3            !
  3            IF NOT .file_error
  3            THEN
  4                BEGIN                               ! Start normal termination
  4
  4                rterm ();
  4                !
U 4 %IF (%BLISS (BLISS32) AND DSRPLUS) %THEN
U 4                IF .gca_pass_count LEQ .gca_rerun_count
U 4                THEN
U 4                    BEGIN
U 4                    ! Doing /AUTOMATIC processing and encountered either cross
U 4                    ! references or a .DO CONTENTS that required more than one
U 4                    ! pass over the input file.
U 4                    !
U 4                    ! Quit now if we have processed the input stream 4 times.
U 4                    ! In this case, we assume that an oscillating cross-refer-
U 4                    ! ence makes it impossible for us to resolve the file
U 4                    ! automatically.
U 4                    !
U 4                    IF .gca_pass_count EQL 4
U 4                    THEN
U 4                        close_i_o_and_return (false, 2)
U 4                    ELSE
U 4                        ! Bump the pass counter and close the input file (the
U 4                        ! output file has already been closed).
U 4                        !
U 4                        BEGIN
U 4                        gca_pass_count = .gca_pass_count + 1;
U 4                        clh (clh_close_input);
U 4                        END;
U 4                    END
U 4                ELSE
U 4                    ! Either /AUTOMATIC was not asserted or all required passes
U 4                    ! over the input file are complete. In any case, close the
U 4                    ! input and output files and return the appropriate status.
  4 %FI
  4                    ! Close output file (do not delete); return either success
  4                    ! or failure.
  5                    BEGIN
```

```
;  551          0678  5                 IF  .gca_fehler
;  552          0679  5                 THEN
;  553          0680  6                     close_i_o_and_return (false, 2)
;  554          0681  5                 ELSE
;  555          0682  5                     close_i_o_and_return (false, 1);
;  556          0683  4                 END;
;  557          0684  4
;  558          0685  4             END                                   ! End normal termination
;  559          0686  3         ELSE
;  560          0687  3             ! Close (don't delete) output file; return 4 (fatal condition).
;  561          0688  3             close_i_o_and_return (false, 4);
;  562          0689  3
;  563          0690  2         END;                                      ! End normal file processing
;  564          0691  2
;  565        U 0692  2 %IF (%BLISS (BLISS32) AND DSRPLUS) %THEN
;  566        U 0693  2         END;                                      ! End of automatic-processing loop
;  567        U 0694  2     true        !avoid compiler complaint
;  568          0695  2 %FI
;  569          0696  1     END;                                  ! End of RUNOFF


                                                                        .TITLE  RUNOFF Overall DSR process control
                                                                        .IDENT  \V04-000\

                                                                        .PSECT  $OWN$,NOEXE,2

                                                     00000 FILE_ERROR:
                                                                        .BLKB   4
                                                     00004 TEMPOB:      .BLKB   244
                                                     000F8 TEMPIB:      .BLKB   244

                                                                        .EXTRN  RNEIOB, RNAIOB, RNOIOB
                                                                        .EXTRN  TTEIOB, FFNAME, SEMCOD
                                                                        .EXTRN  GCA, IRA, IRAC, FS01
                                                                        .EXTRN  LFFCTX, IPFTYP, IPFTOP
                                                                        .EXTRN  KHAR, CLH, DOOPTS
                                                                        .EXTRN  ERM, ERME, ERMS
                                                                        .EXTRN  PUS, RINIT, RTERM
                                                                        .EXTRN  LIB$FIND_FILE, RNFCOI
                                                                        .EXTRN  RNFCOO, RNFFAB, RNFRTL
                                                                        .EXTRN  RNFURE, RNFWFR, LAST_RNF
                                                                        .EXTRN  RNFAAA, IPFTCT

                                                                        .PSECT  $CODE$,NOWRT,2

                                              0FFC 00000                .ENTRY  RUNOFF, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,-   ; 0393
                                                                                R11                                          ;
                            5B      0000'  CF   9E 00002                MOVAB   FILE_ERROR, R11                              ;
                            5A 00000000G   00   9E 00007                MOVAB   FFNAME, R10                                  ;
                            59 00000000G   00   9E 0000E                MOVAB   GCA+208, R9                                  ;
                            58 00000000G   00   9E 00015                MOVAB   CLH, R8                                      ;
                            5E             08   C2 0001C                SUBL2   #8, SP                                       ;
               00000000G    00      00F8   CB   9E 0001F                MOVAB   TEMPIB, RNEIOB                               ; 0448
               00000000G    00        04   AB   9E 00028                MOVAB   TEMPOB, RNAIOB                               ; 0449
                            56 00000000G   00   D0 00030                MOVL    RNAIOB, R6                                   ; 0452
                            57        04   AC   D0 00037                MOVL    RNO_CMD, R7                                  ;
   00F4     8F          00  6E             00   2C 0003B                MOVC5   #0, (SP), #0, #244, (R6)                     ;
```

```
                                          66       00042                                                      ;
                          66 0301003D     8F   D0  00043          MOVL    #50397245, (R6)                      ;
                    04    A6       08     A7   9E  0004A          MOVAB   8(R7), 4(R6)                         ;
                    1E    A6     020E     8F   B0  0004F          MOVW    #526, 30(R6)                         ;
             00000000G    00              00   FB  00055          CALLS   #0, RINIT                            ; 0454
             00000000G    8F 00000000G    8F   D1  0005C          CMPL    #RNFWFR, #LAST_RNF                   ; 0464
                                          11   13  00067          BEQL    1$                                   ;
                                          7E   7C  00069          CLRQ    -(SP)                                ; 0467
                             00000000G    8F   DD  0006B          PUSHL   #RNFAAA                              ;
             00000000G    00              03   FB  00071          CALLS   #3, ERM                              ;
                                          69   11  00078          BRB     3$                                   ; 0468
                          6E 020E0000     8F   D0  0007A  1$:     MOVL    #34471936, $STR$DESC                 ; 0474
                                   04     AE   D4  00081          CLRL    $STR$DESC+4                          ;
                                 4080     8F   BB  00084          PUSHR   #^M<R7,SP>                           ; 0476
                 0000V    CF              02   FB  00088          CALLS   #2, INPUT_FILE_SCAN                  ;
                          04              50   D1  0008D          CMPL    STATUS, #4                           ;
                                          08   12  00090          BNEQ    2$                                   ;
                          50 10000000     E0   9E  00092          MOVAB   268435456(R0), R0                    ; 0477
                                               04  00099          RET                                          ;
                          56 00000000G    00   D0  0009A  2$:     MOVL    RNEIOB, R6                           ; 0479
  00F4   8F          00   6E              00   2C  000A1          MOVC5   #0, (SP), #0, #244, (R6)             ;
                                          66       000A8                                                      ;
                          66 0301003D     8F   D0  000A9          MOVL    #50397245, (R6)                      ;
                    04    A6              6E   9E  000B0          MOVAB   $IOB$FILE_SPEC, 4(R6)                ;
                    1E    A6     020E     8F   B0  000B4          MOVW    #526, 30(R6)                         ;
                                          01   DD  000BA          PUSHL   #1                                   ; 0487
                          68              01   FB  000BC          CALLS   #1, CLH                              ;
                          52              50   D0  000BF          MOVL    R0, CLH_RESULT                       ;
                          01              52   D1  000C2          CMPL    CLH_RESULT, #1                       ; 0489
                                          1F   13  000C5          BEQL    4$                                   ;
                          51              6A   3C  000C7          MOVZWL  FFNAME, SPEC_LENGTH                  ; 0496
                          50       04     AA   D0  000CA          MOVL    FFNAME+4, SPEC_PTR                   ; 0497
                             00000000G    00   DD  000CE          PUSHL   SEMCOD                               ; 0499
                                          03   BB  000D4          PUSHR   #^M<R0,R1>                           ;
                             00000000G    8F   DD  000D6          PUSHL   #RNFCOI                              ;
             00000000G    00              04   FB  000DC          CALLS   #4, ERME                             ;
                                        00D8   31  000E3  3$:     BRW     19$                                  ; 0501
                          05       50     A7   E9  000E6  4$:     BLBC    80(R7), 5$                           ; 0505
                          52              01   D0  000EA          MOVL    #1, CLH_RESULT                       ; 0507
                                          1E   11  000ED          BRB     6$                                   ;
       50       4E   A7   04              06   EF  000EF  5$:     EXTZV   #6, #4, 78(R7), R0                   ; 0514
       69            04   04              50   F0  000F5          INSV    R0, #4, #4, GCA+208                  ;
       50       4F   A7   01              03   EF  000FA          EXTZV   #3, #1, 79(R7), R0                   ; 0515
       69            01   03              50   F0  00100          INSV    R0, #3, #1, GCA+208                  ;
                                          02   DD  00105          PUSHL   #2                                   ; 0516
                          68              01   FB  00107          CALLS   #1, CLH                              ;
                          52              50   D0  0010A          MOVL    R0, CLH_RESULT                       ;
                          01              52   D1  0010D  6$:     CMPL    CLH_RESULT, #1                       ; 0537
                                          20   13  00110          BEQL    7$                                   ;
                          51              6A   3C  00112          MOVZWL  FFNAME, SPEC_LENGTH                  ; 0544
                          50       04     AA   D0  00115          MOVL    FFNAME+4, SPEC_PTR                   ; 0545
                             00000000G    00   DD  00119          PUSHL   SEMCOD                               ; 0549
                                          03   BB  0011F          PUSHR   #^M<R0,R1>                           ;
                             00000000G    8F   DD  00121          PUSHL   #RNFCOO                              ;
             00000000G    00              04   FB  00127          CALLS   #4, ERME                             ;
                                          03   DD  0012E          PUSHL   #3                                   ; 0550
                                          17   11  00130          BRB     8$                                   ;
```

```
                                     57  DD 00132 7$:     PUSHL   R7                                  ; 0603
              00000000G  00          01  FB 00134         CALLS   #1, DOOPTS                          ;
                         0D          50  E8 0013B         BLBS    R0, 9$                              ;
                                     03  DD 0013E         PUSHL   #3                                  ; 0607
                         68          01  FB 00140         CALLS   #1, CLH                             ;
                         77      50  A7  E8 00143         BLBS    80(R7), 19$                         ;
                                     0C  DD 00147         PUSHL   #12                                 ;
                                     70  11 00149 8$:     BRB     18$                                 ;
                         50 00000000G 00 D0 0014B 9$:     MOVL    RNOIOB, R0                          ; 0616
            08       32  A0          04  E1 00152         BBC     #4, 50(R0), 10$                     ;
            06 00000000G 00          04  E0 00157         BBS     #4, TTEIOB+50, 11$                  ; 0617
                     F8  A9    FF64  C9  D0 0015F 10$:    MOVL    GCA+52, GCA+200                     ; 0619
                         50          01  D0 00165 11$:    MOVL    #1, KEEP_READING                    ; 0631
                                     6B  D4 00168         CLRL    FILE_ERROR                          ; 0632
                         09          50  E9 0016A 12$:    BLBC    KEEP_READING, 13$                   ; 0634
                                     5B  DD 0016D         PUSHL   R11                                 ; 0635
                  0000V  CF          01  FB 0016F         CALLS   #1, INPUT_LOOP                      ;
                                     F4  11 00174         BRB     12$                                 ;
                         37          6B  E8 00176 13$:    BLBS    FILE_ERROR, 17$                     ; 0639
              00000000G  00          00  FB 00179         CALLS   #0, RTERM                           ; 0643
                         16      F0  A9  E9 00180         BLBC    GCA+192, 15$                        ; 0678
                                     03  DD 00184         PUSHL   #3                                  ; 0680
                         68          01  FB 00186         CALLS   #1, CLH                             ;
                         05      50  A7  E8 00189         BLBS    80(R7), 14$                         ;
                                     04  DD 0018D         PUSHL   #4                                  ;
                         68          01  FB 0018F         CALLS   #1, CLH                             ;
                         50 10000002 8F  D0 00192 14$:    MOVL    #268435458, R0                      ; 0682
                                         04 00199         RET                                         ;
                                     03  DD 0019A 15$:    PUSHL   #3                                  ;
                         68          01  FB 0019C         CALLS   #1, CLH                             ;
                         05      50  A7  E8 0019F         BLBS    80(R7), 16$                         ;
                                     04  DD 001A3         PUSHL   #4                                  ;
                         68          01  FB 001A5         CALLS   #1, CLH                             ;
                         50 10000001 8F  D0 001A8 16$:    MOVL    #268435457, R0                      ;
                                         04 001AF         RET                                         ; 0688
                                     03  DD 001B0 17$:    PUSHL   #3                                  ;
                         68          01  FB 001B2         CALLS   #1, CLH                             ;
                         05      50  A7  E8 001B5         BLBS    80(R7), 19$                         ;
                                     04  DD 001B9         PUSHL   #4                                  ;
                         68          01  FB 001BB 18$:    CALLS   #1, CLH                             ;
                         50 10000004 8F  D0 001BE 19$:    MOVL    #268435460, R0                      ;
                                         04 001C5         RET                                         ; 0696

; Routine Size:  454 bytes,    Routine Base:  $CODE$ + 0000
;  571      0697  1 GLOBAL ROUTINE grab_resultant (
;  572      0698  1                 function_code          ! For failure action routine
;  573      0699  1                ,primary_code           ! 1st completion code
;  574      0700  1                ,secondary_code         ! 2nd completion code
;  575      0701  1                ,iob : REF $XPO_IOB ()  ! Address of IOB
;  576      0702  1                               ) =
;  577      0703  1 !++
;  578      0704  1 ! FUNCTIONAL DESCRIPTION:
;  579      0705  1 !
;  580      0706  1 !       Moves resultant filename string from the IOB to FFNAME so it
;  581      0707  1 !       can be picked up and used when the error is signaled.
;  582      0708  1 !
;  583      0709  1 ! FORMAL PARAMETERS:
;  584      0710  1 !
;  585      0711  1 !       As defined by XPORT.  See above ROUTINE declaration for the list and
;  586      0712  1 !       meanings of the current formal parameters.
;  587      0713  1 !
;  588      0714  1 ! IMPLICIT INPUTS:      None
;  589      0715  1 !
;  590      0716  1 ! IMPLICIT OUTPUTS:
;  591      0717  1 !
;  592      0718  1 !       FFNAME  String descriptor defined in GLBDAT.BLI that points to the
;  593      0719  1 !               resultant filename string from the IOB pointed to by the
;  594      0720  1 !               fourth parameter passed to this routine.
;  595      0721  1 !
;  596      0722  1 ! ROUTINE VALUE:
;  597      0723  1 ! COMPLETION CODES:
;  598      0724  1 !
;  599      0725  1 !       The PRIMARY_CODE passed to this routine will also be used as its
;  600      0726  1 !       completion code.
;  601      0727  1 !
;  602      0728  1 ! SIDE EFFECTS:         None
;  603      0729  1 !--
;  604      0730  1
;  605      0731  2     BEGIN                                       ! Start grab_resultant
;  606      0732  2         $STR_DESC_INIT (DESCRIPTOR = ffname, CLASS = dynamic);
;  607      0733  2         !
;  608      0734  2         ! Pick of the name and length of the file spec.
;  609      0735  2         !
;  610      0736  2         $STR_COPY ( STRING = iob[iob$t_resultant] ,TARGET = ffname );
;  611      0737  2         semcod = .primary_code;             ! Pick up and save the error code
;  612      0738  2                                             ! explaining why this file couldn't
;  613      0739  2                                             ! be opened.
;  614      0740  2         .primary_code
;  615      0741  1     END;                                    ! End of GRAB_RESULTANT



                                                            .EXTRN  XST$COPY, STR$FAILURE

                                          0004 00000        .ENTRY  GRAB_RESULTANT, Save R2                       ; 0697
                   52 00000000G     00   9E 00002           MOVAB   $STR$DESC, R2                                 ;
                   62 020E0000      8F   D0 00009           MOVL    #34471936, $STR$DESC                          ; 0732
                               04   A2   D4 00010           CLRL    $STR$DESC+4                                   ;
              50        10   AC     1C   C1 00013           ADDL3   #28, IOB, R0                                  ; 0736
                      00000000G     EF   9F 00018           PUSHAB  STR$FAILURE                                   ;
                                    7E   D4 0001E           CLRL    -(SP)                                         ;
```

```
                                               05  BB 00020         PUSHR   #^M<R0,R2>                          ;
                                               7E  D4 00022         CLRL    -(SP)                               ;
                          00000000G  EF        05  FB 00024         CALLS   #5, XST$COPY                        ;
                          00000000G  00    08  AC  D0 0002B         MOVL    PRIMARY_CODE, SEMCOD                ; 0737
                                     50    08  AC  D0 00033         MOVL    PRIMARY_CODE, R0                    ; 0741
                                               04  00037            RET                                         ;

; Routine Size:  56 bytes,    Routine Base:  $CODE$ + 01C6


;  616         U 0742  1 %IF DSRPLUS  %THEN
```

```
ROUTINE init_files (rno_cmd) : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!       INIT_FILES processes either or both of the initialization files
!       that may be present -- DSR$INIT and DSRPLUS$INIT. It attempts to
!       open the files in order and, if found, takes input from them before
!       the main input file is read.
!
! FORMAL PARAMETERS:
!
!       RNO_CMD is the address of the command-line block, passed down from
!       RUNOFF.
!
! IMPLICIT INPUTS:      None
!
! IMPLICIT OUTPUTS:     None
!
! ROUTINE VALUE:
! COMPLETION CODES:     None
!
! SIDE EFFECTS:         None
!--

    BEGIN                                               ! Start of init_files
    MAP
        rno_cmd : REF $rno_cmd;
    LOCAL
        clh_result,
        dsrplus_init,
        keep_reading,
        ptr;

    dsrplus_init = false;
    !
    ! Save the original input file.
    !
    clh (clh_push);
    !
    ! Try to open logical file DSRPLUS$INIT. CLH expects to find the filespec in fs01.
    !
    ptr = CH$PTR (UPLIT (%IF %BLISS (BLISS32) %THEN 'DSRPLUS$INIT'
                         %ELSE                      'DSRPLUS.INIT' %FI ));
    fs_init  (fs01);
    !
    INCR i FROM 1 TO 12 DO
        fs_wchar (fs01, CH$RCHAR_A (ptr));
    !
    clh_result = clh (clh_open_init);
    !
    ! If CLH can open file, save it and try to get next one.
    !
    IF  .clh_result EQL clh_normal
    THEN
        BEGIN
        clh (clh_push);
    dsrplus_init = true;
    END;
!
! Try to open logical file DSR$INIT. CLH expects to find the filespec in fs01.
!
ptr = CH$PTR (UPLIT (%IF %BLISS (BLISS32) %THEN 'DSR$INIT'
                     %ELSE                        'RUNOFF.INIT' %FI ));
!
fs_init  (fs01);
INCR i FROM 1 TO 8 DO
    fs_wchar (fs01, CH$RCHAR_A (ptr));
!
clh_result = clh (clh_open_init);
!
! If CLH can open file, start reading from it. Otherwise, try next one.
!
IF  .clh_result EQL clh_normal
THEN
    BEGIN                                     ! Start processing DSR$INIT file.

    keep_reading = true;
    init_file_error = false;

    WHILE .keep_reading DO
        keep_reading = input_loop (init_file_error);
    !
    ! Close initialization file. (Same as closing .REQUIRE file.)
    !
    clh (clh_close_req);
    !
    ! If an error occurred, recover and close the original source file
    ! and the output "file".
    !
    IF  .init_file_error
    THEN
        BEGIN                                 ! Start pop DSRPLUS$INIT
        !
        ! Pop DSRPLUS$INIT if it was pushed.
        !
        IF  .dsrplus_init THEN
            clh (clh_pop);
        !
        clh (clh_pop);                  ! Recover and close original source file.
        !
        ! Close (don't delete) output file; return 4.
        close_i_o_and_return (false, 4);
        !
        END;                                  ! End pop DSRPLUS$INIT

    END;                                      ! End processing DSR$INIT file

IF  .dsrplus_init
THEN
    BEGIN                                     ! Start processing DSRPLUS$INIT file
    clh (clh_pop);

    keep_reading = true;
```

```
        init_file_error = false;

        WHILE .keep_reading DO
            keep_reading = input_loop (init_file_error);
        !
        ! Close initialization file.
        !
        clh (clh_close_req);
        !
        ! If an error occurred, recover and close original source file
        ! and output destination.
        !
        IF  .init_file_error
        THEN
            BEGIN
            !
            clh (clh_pop);                  ! Recover and close original source file.
            !
            ! Close (don't delete) output file; return 4.
            close_i_o_and_return (false, 4);
            !
            END;

        END;                                        ! End processing DSRPLUS$INIT file
    !
    ! Recover original source file.
    !
    clh (clh_pop);

    END;                                    ! End of INIT_FILES
%FI
```

```
1 ROUTINE input_loop (error_condition) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       INPUT_LOOP reads and processes a single record from the input file.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       ERROR_CONDITION is set to TRUE before return if a file-processing
1 !       error occurred.
1 !
1 ! IMPLICIT INPUTS:      None
1 !
1 ! IMPLICIT OUTPUTS:     None
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       The routine returns TRUE while there is more input to read; it
1 !       returns FALSE when end-of-file is reached, and also when a file
1 !       error occurs. (in this latter case, the formal ERROR_CONDITION is
1 !       also set.)
1 !
1 ! SIDE EFFECTS:         None
1 !--
1
2     BEGIN                                     ! Start of input_loop
2     LOCAL
2         clh_result;
2
2     clh_result = clh (clh_read_input);
2
2     SELECT .clh_result OF
2         SET
2
2         [clh_normal] :          ! Something was read successfully.
2             !
3             BEGIN
3             kcns ();
3             pus  (true);
2             END;
2
2         [clh_end_file] :        ! End of initialization file.
2             !
2             RETURN false;
2
2         [clh_too_long] :        ! Record would not fit in input area (ira).
2             !
3             BEGIN
3             erms (rnfrtl, .fs_start (ira), .fs_length (ira));
3             kcns ();
3             pus  ();
2             END;
2
2         [otherwise] :           ! Unrecoverable error. Inform user.
2             !
```

```
;  821          0945  3                  BEGIN
;  822          0946  3                  erm (rnfure, 0, 0);
;  823          0947  3                  erm (rnffab, 0, 0);
;  824          0948  3                  .error_condition = true;
;  825          0949  3                  RETURN false;
;  826          0950  2                  END;
;  827          0951  2
;  828          0952  2              TES;
;  829          0953  2
;  830          0954  2          RETURN true;
;  831          0955  2
;  832          0956  1          END;                                 ! End of INPUT_LOOP


                                                                              .EXTRN  RINTES

                                                          00FC 00000 INPUT_LOOP:
                                                                              .WORD   Save R2,R3,R4,R5,R6,R7                          ; 0888
                                  57 00000000G   00   9E 00002                MOVAB   ERM, R7                                         ;
                                  56 00000000G   00   9E 00009                MOVAB   PUS, R6                                         ;
                                  55 00000000G   00   9E 00010                MOVAB   KHAR, R5                                        ;
                                  54 00000000G   00   9E 00017                MOVAB   IRA+12, R4                                      ;
                                                 05   DD 0001E                PUSHL   #5                                              ; 0919
                       00000000G  00             01   FB 00020                CALLS   #1, CLH                                         ;
                                  53             50   D0 00027                MOVL    R0, CLH_RESULT                                  ;
                                  52             01   D0 0002A                MOVL    #1, R2                                          ; 0921
                                  01             53   D1 0002D                CMPL    CLH_RESULT, #1                                  ; 0924
                                                 20   12 00030                BNEQ    3$                                              ;
                                                 52   D4 00032                CLRL    R2                                              ;
                                                 64   D5 00034                TSTL    IRA+12                                          ; 0927
                                                 09   14 00036                BGTR    1$                                              ;
                                  65         00G 8F   9A 00038                MOVZBL  #RINTES, KHAR                                   ;
                                  64             01   CE 0003C                MNEGL   #1, IRA+12                                      ;
                                                 0C   11 0003F                BRB     2$                                              ;
                                  50          F8 A4   D0 00041 1$:            MOVL    IRA+4, R0                                       ;
                                  65             60   9A 00045                MOVZBL  (R0), KHAR                                      ;
                                              F8 A4   D6 00048                INCL    IRA+4                                           ;
                                                 64   D7 0004B                DECL    IRA+12                                          ;
                                                 01   DD 0004D 2$:            PUSHL   #1                                              ; 0928
                                  66             01   FB 0004F                CALLS   #1, PUS                                         ;
                                  02             53   D1 00052 3$:            CMPL    CLH_RESULT, #2                                  ; 0931
                                                 04   12 00055                BNEQ    4$                                              ;
                                                 52   D4 00057                CLRL    R2                                              ;
                                                 58   11 00059                BRB     9$                                              ; 0933
                                  04             53   D1 0005B 4$:            CMPL    CLH_RESULT, #4                                  ; 0935
                                                 30   12 0005E                BNEQ    7$                                              ;
                                                 52   D4 00060                CLRL    R2                                              ;
                                                 64   DD 00062                PUSHL   IRA+12                                          ; 0938
                                              F4 A4   DD 00064                PUSHL   IRA                                             ;
                                       00000000G 8F   DD 00067                PUSHL   #RNFRTL                                         ;
                       00000000G  00             03   FB 0006D                CALLS   #3, ERMS                                        ;
                                                 64   D5 00074                TSTL    IRA+12                                          ; 0939
                                                 09   14 00076                BGTR    5$                                              ;
                                  65         00G 8F   9A 00078                MOVZBL  #RINTES, KHAR                                   ;
                                  64             01   CE 0007C                MNEGL   #1, IRA+12                                      ;
                                                 0C   11 0007F                BRB     6$                                              ;
```

```
                              50        F8   A4  D0 00081 5$:     MOVL     IRA+4, R0                              ;
                              65             60  9A 00085         MOVZBL   (R0), KHAR                             ;
                                        F8   A4  D6 00088         INCL     IRA+4                                  ;
                                             64  D7 0008B         DECL     IRA+12                                 ;
                              66             00  FB 0008D 6$:     CALLS    #0, PUS                                ; 0940
                              1C             52  E9 00090 7$:     BLBC     R2, 8$                                 ; 0943
                                             7E  7C 00093         CLRQ     -(SP)                                  ; 0946
                                 00000000G   8F  DD 00095         PUSHL    #RNFURE                                ;
                              67             03  FB 0009B         CALLS    #3, ERM                                ;
                                             7E  7C 0009E         CLRQ     -(SP)                                  ; 0947
                                 00000000G   8F  DD 000A0         PUSHL    #RNFFAB                                ;
                              67             03  FB 000A6         CALLS    #3, ERM                                ;
                          04  BC             01  D0 000A9         MOVL     #1, @ERROR_CONDITION                   ; 0948
                                             04  11 000AD         BRB      9$                                     ; 0949
                              50             01  D0 000AF 8$:     MOVL     #1, R0                                 ; 0954
                                                 04 000B2         RET                                             ;
                                             50  D4 000B3 9$:     CLRL     R0                                     ; 0956
                                                 04 000B5         RET                                             ;

; Routine Size:  182 bytes,    Routine Base:  $CODE$ + 01FE


;  833          0957  1
```

```
1 %IF %BLISS(BLISS32) %THEN
1 ROUTINE input_file_scan ( rno_cmd
1                          ,input_file_name : $STR_DESCRIPTOR (CLASS = dynamic)
1                          ) =
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine makes the needed calls to the VMS routine LIB$FIND_FILE.
1 !       Also, for DSRPLUS, it does the file type resolution.  What this means
1 !       is that if the user didn't specify an extension and the first FIND_FILE
1 !       doesn't locate a file with a .RN_ extension, then we will try our own
1 !       list of extensions starting with .RNO.  Data structure IPFTOP contains
1 !       the list of known extensions in the order in which we will look for
1 !       them.  But, for the sake of documentation, here is that list:
1 !
1 !               Input file      Maps to this output
1 !               filetype        file filetype
1 !
1 !               .RNO                    .MEM
1 !               .RTB                    .MET            (DSRPLUS only)
1 !               .RNT                    .MEC
1 !               .RNX                    .MEX
1 !               .RND                    .DOC
1 !               .RNH                    .HLP
1 !               .RNB                    .BLB
1 !               .RNC                    .CCO
1 !               .RNE                    .ERR
1 !               .RNL                    .PLM
1 !               .RNM                    .MAN
1 !               .RNP                    .OPR
1 !               .RNS                    .STD
1 !               .***                    Wild card  (DSRPLUS only)
1 !
1 !       Note that this routine is conditionalized to be compiled only if
1 !       the VMS implementation is being built.  As for the TOPS-20 implementa-
1 !       tion, the input and output file extensions are defaulted within the
1 !       command line interface in RNOT20.  However, this will not take care
1 !       of the TOPS-10 (or any future PDP-11) implementation.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       RNO_CMD         Data structure from the CLI.  This is where we will
1 !                       get the filespec string that the user entered.
1 !
1 !       INPUT_FILE_NAME Address of a string descriptor into which we will
1 !                       put the filespec resulting from our efforts.
1 !
1 ! IMPLICIT INPUTS:      None
1 !
1 ! IMPLICIT OUTPUTS:     None
1 !
1 ! ROUTINE VALUE:
1 !
1 !       1 = Normal return
1 !       4 = Fatal return -- file not found
1 !
1 ! SIDE EFFECTS:         None
```

```
  1 !--
  1
  2         BEGIN                                   ! Start of input_loop
  2         MAP
  2             rno_cmd : REF $rno_cmd;
  2
  2         BIND
  2             passed_file_spec = rno_cmd [rno$t_input] : $STR_DESCRIPTOR(),
  2             ip_file_spec = .input_file_name : $STR_DESCRIPTOR();
  2
  2         LOCAL
  2             rms_status,
  2             rms_stv_adr : INITIAL(-1),
  2             rms_user_flags : INITIAL(2),
  2             default_file_specification : $STR_DESCRIPTOR (CLASS = dynamic);
  2
  2         default_file_specification [str$b_dtype] = str$k_dtype_t;    ! ASCII text (8-bit)
  2         default_file_specification [str$b_class] = str$k_class_f;    ! Fixed (Scalar) String Descriptor
  2         default_file_specification [STR$H_LENGTH] = 4;
  2         default_file_specification [STR$A_POINTER] = .ipftop[0];
  2
  2         ipftyp = -1;          ! To indicate that we haven't mapped against IPFTOP
  2
  2         rms_status = LIB$FIND_FILE ( passed_file_spec
  2                                     ,.input_file_name
  2                                     ,lffctx
  2                                     ,default_file_specification
  2                                     ,0                  ! No related file specification
  2                                     ,rms_stv_adr        ! Secondary RMS error code
  2                                     ,rms_user_flags     ! User flags
  2                                     );
  2
U 2 %IF dsrplus %THEN
U 2     IF .rms_status EQL RMS$_FNF                          ! If a file wasn't
U 2                 AND                                      !   found and a
U 2     CH$FAIL( CH$FIND_CH( .passed_file_spec[str$h_length] !  filetype was not
U 2                         ,.passed_file_spec[str$a_pointer] ! specified,
U 2                         ,%C'.' ) )
U 2     THEN
U 2         BEGIN                           ! We will try our own default extension(s).
U 2
U 2 !!!     lffctx = 0;         ! To start searching all over again.
U 2
U 2         ! Search all input filetypes until we find one:
U 2
U 2         INCR ipftyp FROM 0 TO (ipftct - 1) DO
U 2             BEGIN
U 2
U 2             ! Point to the currently indexed input filetype option.  The
U 2             ! length field has already been initialized to 4 outside of
U 2             ! this INCR loop.
U 2
U 2             default_file_specification [STR$A_POINTER] = .ipftop[.ipftyp];
U 2
U 2             ! Now see if there is a file with the specified filename and
U 2             ! the currently indexed input filetype option.
U 2
```

```
;  949      U 1072  2               rms_status = LIB$FIND_FILE ( passed_file_spec
;  950      U 1073  2                                           ,.input_file_name
;  951      U 1074  2                                           ,lffctx
;  952      U 1075  2                                           ,default_file_specification
;  953      U 1076  2                                           ,0        ! No related file specification
;  954      U 1077  2                                           ,rms_stv_adr  ! Secondary RMS error code
;  955      U 1078  2                                           ,2                    ! User flags
;  956      U 1079  2                                           );
;  957      U 1080  2
;  958      U 1081  2               IF .rms_status EQL RMS$_NORMAL THEN EXITLOOP
;  959      U 1082  2               END                     !end of INCR loop.
;  960      U 1083  2           END;            ! End of what we do if first test didn't find a file
;  961        1084  2 %FI
;  962        1085  2
;  963        1086  2       IF  .rms_status EQL RMS$_NORMAL         ! Succeeded in finding (any type)?
;  964        1087  2       THEN
;  965        1088  3           BEGIN
;  966        1089  3
;  967        1090  3           ! Pick off the name and length of the file spec for ERROR.BLI
;  968        1091  3           irac_fspecp = .ip_file_spec [STR$A_POINTER];
;  969        1092  3           irac_fspecc = .ip_file_spec [STR$H_LENGTH];
;  970      U 1093  3 %IF dsrplus %THEN
;  971      U 1094  3           IF  (.ipftyp GTR 0)  ! If we didn't find an RNO file,  tell the
;  972      U 1095  3           THEN            ! user just what we did find.
;  973      U 1096  3               erm (rnfoft); ! File type ".RNO" not found.  Processing file xx
;  974        1097  3 %FI
;  975        1098  3           true            ! Exit saying we found an input file.
;  976        1099  3           END
;  977        1100  2       ELSE
;  978        1101  3           BEGIN
;  979        1102  3           LOCAL
;  980        1103  3               spec_length,
;  981        1104  3               spec_ptr;
;  982        1105  3
;  983        1106  3           spec_ptr    = .passed_file_spec [str$a_pointer];
;  984        1107  3           spec_length = .passed_file_spec [str$h_length];
;  985        1108  3
;  986        1109  3           erme (rnfcoi, .spec_ptr, .spec_length, .semcod);
;  987        1110  3           IF .rms_stv_adr EQL -1 THEN
;  988        1111  3               SIGNAL ( .rms_status )
;  989        1112  3           ELSE
;  990        1113  3               SIGNAL ( .rms_status, .rms_stv_adr );
;  991        1114  3
;  992        1115  3           4       ! Exit saying we couldn't find an input file.
;  993        1116  3           END
;  994        1117  1       END;                                ! End of routine INPUT_FILE_SCAN



                                  003C 00000 INPUT_FILE_SCAN:
                                                  .WORD     Save R2,R3,R4,R5                                 ; 0959
              55 00000000G    00   9E 00002       MOVAB     LIB$SIGNAL, R5                                   ;
              5E              08   C2 00009       SUBL2     #8, SP                                           ;
              52         04   AC   7D 0000C       MOVQ      RNO_CMD, R2                                      ; 1022
              7E              01   CE 00010       MNEGL     #1, RMS_STV_ADR                                  ; 1023
```

```
                                         02  DD 00013         PUSHL   #2                                          :
                  08   AE 010E0004       8F  D0 00015         MOVL    #17694724, DEFAULT_FILE_SPECIFICATION      : 1033
                  0C   AE 00000000G      00  D0 0001D         MOVL    IPFTOP, DEFAULT_FILE_SPECIFICATION+4       : 1034
           00000000G   00                01  CE 00025         MNEGL   #1, IPFTYP                                 : 1036
                                         5E  DD 0002C         PUSHL   SP                                         : 1038
                               08        AE  9F 0002E         PUSHAB  RMS_STV_ADR                                :
                                         7E  D4 00031         CLRL    -(SP)                                      :
                               14        AE  9F 00033         PUSHAB  DEFAULT_FILE_SPECIFICATION                 :
                        00000000G        00  9F 00036         PUSHAB  LFFCTX                                     :
                               08        AC  DD 0003C         PUSHL   INPUT_FILE_NAME                            : 1039
                                         52  DD 0003F         PUSHL   R2                                         : 1038
           00000000G   00                07  FB 00041         CALLS   #7, LIB$FIND_FILE                          :
                       54                50  D0 00048         MOVL    R0, RMS_STATUS                             :
           00010001    8F                54  D1 0004B         CMPL    RMS_STATUS, #65537                         : 1086
                                         13  12 00052         BNEQ    1$                                         :
           00000000G   00          04    A3  D0 00054         MOVL    4(R3), IRAC+16                             : 1091
           00000000G   00                63  3C 0005C         MOVZWL  (R3), IRAC+20                              : 1092
                       50                01  D0 00063         MOVL    #1, R0                                     : 1088
                                             04 00066         RET                                                :
                       51          04    A2  D0 00067 1$:     MOVL    4(R2), SPEC_PTR                            : 1106
                       50                62  3C 0006B         MOVZWL  (R2), SPEC_LENGTH                          : 1107
                        00000000G        00  DD 0006E         PUSHL   SEMCOD                                     : 1109
                                         50  DD 00074         PUSHL   SPEC_LENGTH                                :
                                         51  DD 00076         PUSHL   SPEC_PTR                                   :
                        00000000G        8F  DD 00078         PUSHL   #RNFCOI                                    :
           00000000G   00                04  FB 0007E         CALLS   #4, ERME                                   :
           FFFFFFFF    8F          04    AE  D1 00085         CMPL    RMS_STV_ADR, #-1                           : 1110
                                         07  12 0008D         BNEQ    2$                                         :
                                         54  DD 0008F         PUSHL   RMS_STATUS                                 : 1111
                       65                01  FB 00091         CALLS   #1, LIB$SIGNAL                             :
                                         08  11 00094         BRB     3$                                         :
                                   04    AE  DD 00096 2$:     PUSHL   RMS_STV_ADR                                : 1113
                                         54  DD 00099         PUSHL   RMS_STATUS                                 :
                       65                02  FB 0009B         CALLS   #2, LIB$SIGNAL                             :
                       50                04  D0 0009E 3$:     MOVL    #4, R0                                     : 1101
                                             04 000A1         RET                                                : 1117

; Routine Size:  162 bytes,    Routine Base:  $CODE$ + 02B4


;  995          1118  1 %FI
;  996          1119  1 END                                 ! End of module
;  997          1120  0 ELUDOM



                                                             .EXTRN  LIB$SIGNAL

;                         PSECT SUMMARY
;
;       Name                   Bytes                       Attributes
;
;  $OWN$                         492  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        854  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;                               Library Statistics
;
;                                               -------- Symbols --------       Pages           Processing
;       File                                    Total   Loaded  Percent         Mapped          Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1            9776         2        0          581             00:01.0
;  _$255$DUA28:[SYSLIB]XPORT.L32;1               590       117       19          252             00:00.6
;  _$255$DUA28:[RUNOFF.SRC]DSRLIB.L32;1         1248       127       10           86             00:01.0





;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:RUNOFF/OBJ=OBJ$:RUNOFF MSRC$:RUNOFF/UPDATE=(ENH$:RUNOFF)

; Size:           854 code + 492 data bytes
; Run Time:          00:28.3
; Elapsed Time:      00:58.5
; Lines/CPU Min:       2371
; Lexemes/CPU-Min: 42594
; Memory Used:  251 pages
; Compilation Complete
```

0348 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RSDERM
LIS

RTERM
LIS

RNODEF
LIS

RUNOFF
LIS

RNOVMS
LIS

RSKIPS
LIS

RNODAT
LIS

SAVRES
LIS

RNOVRS
LIS

RNFERM
LIS