

Table of contents

2-	1	SPLINI -- Initialization for spooling system
3-	1	CHKSD -- Check for spooled on .ENTER
4-	1	SDMOVE -- Write user's data to spool file
5-	1	SBUFOT -- Write current buffer to file
6-	1	SGETBF -- Reserve buffer for file
7-	1	SFLUSH -- Write current buffer to file
8-	1	SPLSHF -- Set or reset hold flag for file
9-	1	SPOLGO -- Try to start spooled device
10-	1	SPOLRD -- Read next block from spool file
11-	1	SPOOLQ -- Put spooler in wait queue
12-	1	SFRCMP -- Spool read completion routine
13-	1	SPOLWR -- Write to spooled device
14-	1	SWCMP -- Spool write completion routine
15-	1	SWCRLF -- Write cr-lf to spooled device
16-	1	SDFUNC -- Send special function code to spooled device
17-	1	SDCLOS -- Close spooled channel
18-	1	SPLDEL -- Delete a spool file
19-	1	FRESFB -- Release a SFCB
20-	1	GTSBLK -- Get free spool file disk block
21-	1	RLSBLK -- Free spool file disk block
22-	1	RLSBUF -- Release spool buffer
23-	1	FNDSD -- Locate SFCB for spooled channel
24-	1	CVTDVU -- Convert device name to dev index and unit #
25-	1	EMSPHL -- EMT to set spool HOLD mode
26-	1	EMSPNH -- EMT to set spool NOHOLD mode
27-	1	SFLGPG -- COPY FLAG PAGE TO SPOOL BUFFER

```

1          .TITLE  TSSPOL  T/S SPOOLING ROUTINES
2
3          ;
4          ;   COPYRIGHT (C) 1976,1977,1978,1979,1980,1981,1982,1983,1984,1985,
5          ;   1986,1987,1988.
6          ;
7          ;   S&H COMPUTER SYSTEMS, INC.
8          ;   NASHVILLE, TENNESSEE
9          ;
10         ;   THE ROUTINES IN THIS MODULE IMPLEMENT AUTOMATIC
11         ;   SPOOLING TO OUTPUT DEVICES SUCH AS LINE PRINTERS.
12         ;
13         ;   .CSECT  TSSPOL
14         ;   TSSPOL: .RAD50  /SPL/           ;System overlay ID word
15         ;
16         ;   .MCALL  .READW,.WRITW,.PRINT
17         ;
18         ;   .ENABL  LC
19         ;   .DSABL  GBL
20         ;-----
21         ;   MACROS TO ENABLE AND DISABLE INTERRUPTS.
22         ;
23         ;   .MACRO  DISABL
24         ;   BIS     #340,@#PSW
25         ;   .ENDM   DISABL
26         ;
27         ;   .MACRO  ENABL
28         ;   BIC     INTPRI,@#PSW
29         ;   .ENDM   ENABL
30         ;
31         ;   Macro definition to call global routines residing in mapped system regions.
32         ;
33         ;   .MACRO  OCALL  ENTADD
34         ;   .IF      B,ENTADD
35         ;   .ERROR   ;OCALL SPECIFIED WITH NO ENTRY ADDRESS
36         ;   .MEXIT
37         ;   .ENDC
38         ;   CALL     OVRHC           ;CALL THE OVERLAY HANDLER
39         ;   .WORD    ENTADD         ;SPECIFY THE ENTRY POINT
40         ;   .ENDM
41         ;
42         ;   GLOBAL DEFINITIONS
43         ;
44         ;   .GLOBL  CHKSD,SDCLOS,TSSPOL,EMSPHL,EMSPNH
45         ;   .GLOBL  SPOLGO,SPLSHF,SPLINI,SFLGPG
46         ;   .GLOBL  SDMOVE,SPLTOP,SPLDEL
47         ;
48         ;   GLOBAL REFERENCES
49         ;
50         ;   .GLOBL  DF$CLS,DS$NRD,DVSTAT,Q.FUNC,DF$ENT,EMTXIT
51         ;   .GLOBL  EMTBLK,SDCB,SDCBND,NUMDEV,PNAME,DX$NRD,DVFLAG
52         ;   .GLOBL  CORUSR,SPLND,SPLNB,SF$BN1,CSIBUF,CSIBND
53         ;   .GLOBL  SDFRBL,PVSPBL,UFORM,SPDLBF
54         ;   .GLOBL  NFRESB,SB$TXT,QMSG,GTSYMB,LUNAME
55         ;   .GLOBL  SFCBSZ,SF$1ST,SF$DEL,SDNAME,DYDEV
56         ;   .GLOBL  SD$SMS,SD$SNG,SFFORM,NSPLFL,NSPLBL
57         ;   .GLOBL  SDFORM,SDANAM,SD$FLK,SD$WFM,SD$FLG

```

```

58      .GLOBL  $DILUP,LSW,SFID,SPOLID
59      .GLOBL  NESB,SD$BWT,FMMSG,SPLERR,FLGPAG
60      .GLOBL  SD$HLD,SF$HLD,SNBUX,LPROG,LPROJ
61      .GLOBL  SXBPNT,SXSFCB,SDDVU
62      .GLOBL  FILSPC,CS$SPL,SPUBUF,SPLCHN,SPLARG
63      .GLOBL  INTPRI,PSW,CHNADR,C.CSW
64      .GLOBL  S$PCB,S$SPDB,USPLCH
65      .GLOBL  SPLBHD,SFCB,SFCBND,SYBFAD
66      .GLOBL  UREGO,QNSPND,CHKABT
67      .GLOBL  SFUSER,SFFLAG,SFCHAN,SFFILE
68      .GLOBL  SFSDCB,SFNMBL,SFSTRT,SFFLNK
69      .GLOBL  SFQLNK,SF$BSY,SDCHAN,SDFLAG
70      .GLOBL  SDSFCB,SDFLNK,SDBUF1,SDBUF2
71      .GLOBL  CTRLTT,SDWLST,FREUSR
72      .GLOBL  SDBLK,SDFHD,SD$INR
73      .GLOBL  SD$DEL,SDCBSZ,SBUFWD
74      .GLOBL  SFCBFH,SD$CLR,SD$BAK,SD$RSV
75      .GLOBL  SDBU,SDBULS,SDSKIP,LPRG1,LPRG2
76      .GLOBL  CHNNUM,URO,GETUCH,SPUBND
77      .GLOBL  GETSYQ,Q.BLKN
78      .GLOBL  Q.WCNT,Q.COMP,Q.CHAN,SYQID
79      ; Global for mapped system handler
80      .GLOBL  OVRHC
81
82      ;
83      ; LOCAL DATA CELLS
84      ;
85 000002 000000 GRNTBL: .WORD 0 ;Addr of start of spool file allocation table
86 000004 000000 GRNEND: .WORD 0 ;Addr of end of spool file allocation table
87 000006 000000 SPBWH: .WORD 0 ;SPOOL DEVICES WAITING FOR BUFFERS
88 000010 015 012 CRTXT: .BYTE CR,LF
89      ;
90      ; MISC. PARAMETERS.
91      ;
92 000015 CR = 15 ;CARRIAGE RETURN
93 000012 LF = 12 ;LINE FEED
94 000007 BELL = 7 ;BELL
95 000135 RSQBR = 135 ;RIGHT SQUARE BRACKET

```

SPLINI -- Initialization for spooling system

```

1          .SBTTL  SPLINI -- Initialization for spooling system
2          ;-----
3          ; SPLINI is called during system initialization to perform spooler related
4          ; system initialization.
5          ;
6          ;
7 000012    010246    SPLINI: MOV      R2,-(SP)
8 000014    010346    MOV      R3,-(SP)
9 000016    010446    MOV      R4,-(SP)
10         ;
11         ; Allocate spool file table at end of this segment
12         ;
13 000020    012703    005252'    MOV      #SPLTOP,R3      ;Get address of start of buffer area
14 000024    010367    177752    MOV      R3,GRNTBL      ;Set address of start of table
15 000030    016702    000000G    MOV      NSPLBL,R2      ;Get # blocks for spool file
16 000034    062702    000017    ADD      #15.,R2        ;Bound up to word boundary
17 000040    072227    177774    ASH      #-4,R2        ;Divide by 16 to get # words to allocate
18 000044    012700    177777    MOV      #-1,R0        ;Get word with all bits set
19 000050    010023    10$: MOV      R0,(R3)+      ;Set all bits on in table
20 000052    077202    SOB          R2,10$          ;Fill entire table with ones
21 000054    010367    177724    MOV      R3,GRNEND      ;Set address of end of table
22         ;
23         ; Allocate spool buffers within this system overlay region just beyond
24         ; the end of the code.
25         ;
26 000060    012702    000000G    MOV      #SPLNB,R2      ;Get number of spool buffers wanted
27 000064    001414    BEQ          3$              ;Br if none wanted
28 000066    010367    000000G    MOV      R3,SPLBHD      ;Set pointer to start of buffer list
29 000072    000407    BR          2$              ;
30 000074    020327    137000    1$: CMP      R3,#137000    ;Would another buffer overflow segment?
31 000100    103005    BHIS        4$              ;Br if yes
32 000102    010300    MOV      R3,R0              ;Get address of current buffer
33 000104    062703    001000    ADD      #512.,R3      ;Calculate address of next buffer
34 000110    010310    MOV      R3,(R0)            ;Make current buffer point to next buffer
35 000112    077210    2$: SOB          R2,1$          ;Loop for all buffers except last
36 000114    005013    4$: CLR          (R3)          ;Make link in last buffer be zero
37         ;
38         ; Initialize spool file control block linked list
39         ;
40 000116    016702    000000G    3$: MOV      SFCB,R2      ;Point to start of SFCB area
41 000122    010267    000000G    MOV      R2,SFCBFH      ;Set this as the 1st free SFCB
42 000126    016703    000000G    MOV      NSPLFL,R3      ;Get # of SFCB's
43 000132    010204    MOV      R2,R4              ;Point to next SFCB
44 000134    012700    000000C    6$: MOV      #SFCBSZ/2,R0  ;Get # words in SFCB
45 000140    005024    8$: CLR      (R4)+          ;Zero the SFCB
46 000142    077002    SOB          R0,8$          ;
47 000144    020327    000001    CMP      R3,#1        ;Is this the last SFCB?
48 000150    001404    BEQ          5$              ;Br if yes
49 000152    010462    000000G    MOV      R4,SFQLNK(R2)    ;Set forward link to next SFCB
50 000156    010402    MOV      R4,R2              ;Point to next SFCB
51 000160    077313    SOB          R3,6$          ;Loop of more to initialize
52         ;
53         ; Initialize word that contains number of free public spool file blocks
54         ;
55 000162    016702    000000G    5$: MOV      NSPLBL,R2      ;Get total number of spool file blocks
56 000166    162702    000000C    SUB      #<SPLND*<SNBUX+10.>>,R2 ;Subtract number of private blocks
57 000172    010267    000000G    MOV      R2,NFRESB      ;Set number of free public blocks

```

```
58 ;  
59 ; Finished  
60 ;  
61 000176 012604 9$: MOV (SP)+, R4  
62 000200 012603 MOV (SP)+, R3  
63 000202 012602 MOV (SP)+, R2  
64 000204 000207 RETURN
```

CHKSD -- Check for spooled on .ENTER

```

1          .SBTTL  CHKSD  -- Check for spooled on .ENTER
2
3          ; -----
4          ;  CHKSD IS CALLED FOR EACH .ENTER TO SEE IF THE DEVICE
5          ;  TO BE OPENED IS A SPOOLED DEVICE.  IF THE .ENTER IS FOR
6          ;  A SPOOLED DEVICE, A SPOOL FILE CONTROL BLOCK (SFCB) IS
7          ;  SET UP AND LINKED INTO THE WAITING QUEUE FOR THE REQUESTED
8          ;  DEVICE.  SEE TSDEFS FOR THE FORMAT OF A SFCB AND SDCB.
9          ;  WHEN CALLED THE ARGUMENT LIST FOR THE .ENTER MUST BE IN
10         ;  EMTBLK.
11         ;  ALL REGISTERS ARE PRESERVED.
12
13         ;  Inputs:
14         ;  FILSPC = File specification
15         ;  CHNADR = Address of channel block.
16         ;  CHNUM = Current channel number
17
18         ;  Outputs:
19         ;  C-flag cleared ==> This is a spooled device.
20         ;  C-flag set    ==> This is not a spooled device.
21
22         CHKSD:  MOV     R0,-(SP)
23                MOV     R1,-(SP)
24                MOV     R3,-(SP)
25                MOV     FILSPC,R1      ;GET DEVICE NAME
26                CALL    CVTDVU        ;Convert to dev index # and unit #
27                MOV     #SDCB,R0      ;POINT TO FIRST SDCB
28                2$:     CMP     R0,#SDCBND      ;CHECKED ALL SDCB'S?
29                BHIS    5$            ;BRANCH IF YES
30                CMP     SDNAME(R0),#DMYDEV; Is this an entry for dummy device?
31                BEQ     13$          ;Br if yes
32                CMP     R1,SDDVU(R0)   ;DO NAMES MATCH?
33                BEQ     3$            ;BRANCH IF YES
34                13$:    ADD     #SDCBSZ,R0      ;GO CHECK NEXT SDCB
35                BR      2$
36
37         ;  Channel is being opened to a spooled device.
38         ;  Save spool file name on stack then free USR module.
39
40                3$:     MOV     FILSPC+2,-(SP) ;SAVE 1ST 3 CHARS OF FILE NAME
41                MOV     FILSPC+4,-(SP) ;SAVE LAST 3 CHARS OF FILE NAME
42                MOV     R0,-(SP)      ;PROTECT SDCB ADDRESS
43                OCALL    FREUSR        ;RELEASE OUR LOCK ON USR MODULE
44                MOV     (SP)+,R0      ;RECOVER SDCB ADDRESS
45                BR      1$
46
47         ;  THERE ARE NO FREE SFCB'S.
48         ;  SUSPEND USER UNTIL ONE IS RELEASED.
49                4$:     MOV     R0,-(SP)      ;SAVE SDCB ADDRESS
50                MOV     #S$PCB,R0      ;QUEUE FOR CONTROL BLOCK
51                CALL    QNSPND        ;ENQ AND SUSPEND EXECUTION
52                CALL    CHKABT        ;SEE IF WE WERE ABORTED WHILE ASLEEP
53                MOV     (SP)+,R0      ;RECOVER SDCB ADDRESS
54
55         ;  FOUND SPOOLED DEVICE -- SET UP SFCB.
56         ;  FIND FREE SFCB.
57                1$:     DISABL                    ;** DISABLE **
58                MOV     SFCBFH,R1      ;HEAD OF FREE SFCB CHAIN
59                BEQ     4$            ;BRANCH IF NONE FREE
60                MOV     SFQLNK(R1),SFCBFH;REMOVE SFCB FROM LIST

```

```

58 000350          ENABL                      ;** ENABLE **
59                ; GOT FREE SFCB (R1) -- SET IT UP
60 000356 116761 000000G 000000G          MOVB  CORUSR,SFUSER(R1);SET USER # IN SFCB
61 000364 105061 000000G          CLRB   SFFLAG(R1) ;CLEAR CONTROL FLAGS
62                ; Assign unique ID number to spool file
63 000370 026727 000000G 023417          CMP    SPOLID,#9999. ;Is spool file ID number about to overflow?
64 000376 103402          BLO     11$        ;Br if not
65 000400 005067 000000G          CLR     SPOLID ;Reset spool ID number
66 000404 005267 000000G 11$: INC     SPOLID ;Increment spool file ID number
67 000410 016761 000000G 000000G          MOV    SPOLID,SFID(R1) ;Set ID number for this file
68                ; Save file name in SFCB
69 000416 012661 000002G          MOV     (SP)+,SFFILE+2(R1);SET LAST 3 CHARS OF FILE NAME
70 000422 012661 000000G          MOV     (SP)+,SFFILE(R1);SET 1ST 3 CHARS OF FILE NAME
71 000426 001010          BNE     10$        ;BR IF FILE NAME WAS SPECIFIED
72 000430 116703 000000G 9$: MOVB  CORUSR,R3 ;GET JOB INDEX NUMBER
73 000434 016361 000000G 000000G          MOV    LPRG1(R3),SFFILE(R1);IF NO FILE NAME SPECIFIED, USE
74 000442 016361 000000G 000002G          MOV    LPRG2(R3),SFFILE+2(R1);PROGRAM NAME
75                ; SET DEFAULT FORM NAME IN SFCB
76                ; (USE LAST FORM NAME SPECIFIED BY "FORM" COMMAND)
77 000450 016761 000000G 000000G 10$: MOV    UFORM,SFFORM(R1)
78 000456 016761 000002G 000002G          MOV    UFORM+2,SFFORM+2(R1)
79 000464 016761 000004G 000004G          MOV    UFORM+4,SFFORM+4(R1)
80                ; SEE IF OUTPUT SHOULD BE HELD UNTIL CHANNEL IS CLOSED.
81 000472 032760 000000G 000000G          BIT     #SD$HLD,SDFLAG(R0);IS DEVICE IN HOLD MODE?
82 000500 001403          BEQ     8$        ;BR IF NOT
83 000502 152761 000000G 000000G          BISB  #SF$HLD,SFFLAG(R1);SET HOLD FLAG IN FILE
84 000510 116761 000000G 000000G 8$: MOVB  CHNNUM,SFCHAN(R1);SAVE CHANNEL #
85 000516 010061 000000G          MOV     R0,SFSDCB(R1) ;ADDRESS OF SDCB
86 000522 005061 000000G          CLR     SFNMBL(R1) ;# OF BLOCKS IN FILE
87                ; ADD SFCB TO TAIL OF LIST OF FILES WAITING FOR DEVICE
88 000526          DISABL                      ;** DISABLE **
89 000534 062700 000000G          ADD     #SDFHD,R0 ;POINT TO LIST HEAD IN SDCB
90 000540 005710 7$: TST     (R0) ;AT END OF LIST?
91 000542 001404          BEQ     6$        ;BRANCH IF YES
92 000544 011000          MOV     (R0),R0 ;CHAIN TO NEXT SFCB
93 000546 062700 000000G          ADD     #SFQLNK,R0 ;POINT TO ITS FLINK
94 000552 000772          BR      7$
95 000554 010110 6$: MOV     R1,(R0) ;ADD US TO END OF CHAIN
96 000556 005061 000000G          CLR     SFQLNK(R1) ;CLEAR OUR LIST FLINK
97 000562          ENABL                      ;** ENABLE **
98                ; ALLOCATE FIRST DISK BLOCK FOR FILE
99 000570 004767 003510          CALL    GTSBLK ;GET A DISK BLOCK
100 000574 010061 000000G          MOV     R0,SFSTRT(R1) ;SET AS START ADDR OF FILE
101 000600 010061 000000G          MOV     R0,SFFLNK(R1) ;SET AS NEXT BLOCK ADDR
102                ;
103                ; Set up channel so it will do I/O to spool file.
104                ;
105 000604 016703 000000G          MOV     CHNADR,R3 ;GET ADDRESS OF CHANNEL BLOCK
106 000610 052763 000000G 000000G          BIS   #CS$SPL,C.CSW(R3);SET FLAG SAYING THIS IS A SPOOLED CHANNEL
107 000616 000241          CLC                      ;Signal success on return
108 000620 000401          BR      12$
109                ;
110                ; This is not a spooled device
111                ;
112 000622 000261 5$: SEC                      ;Signal failure on return
113                ;
114                ; Finished

```

```
115  
116 000624 012603      ;  
117 000626 012601      12$:  MOV      (SP)+, R3  
118 000630 012600      MOV      (SP)+, R1  
119 000632 000207      MOV      (SP)+, R0  
                        RETURN
```

SDMOVE -- Write user's data to spool file

```

1          .SBTTL  SDMOVE -- Write user's data to spool file
2          ;-----
3          ; SDMOVE is called during .write emt processing to move the user's data
4          ; from his buffer to the spool buffer and then write it to the spool file.
5          ;
6          ; Inputs:
7          ;   CHNNUM = User's channel number.
8          ;   EMTBLK = EMT argument block.
9          ;
10         SDMOVE: MOV     R1,-(SP)
11                 MOV     R2,-(SP)
12                 MOV     R3,-(SP)
13                 MOV     R4,-(SP)
14                 MOV     R5,-(SP)
15         ;
16         ; Find SDCB associated with this channel
17         ;
18         000646 004767 004050      CALL     FNDSB          ;FIND SDCB FOR THIS CHANNEL
19         000652 103534             BCS      SDMVX          ;BR IF COULDN'T FIND IT
20         ;
21         ; Make sure spool file buffer is assigned to our channel
22         ;
23         000654 004767 000352      B$:      CALL     SGETBF          ;GET SPOOL FILE BUFFER FOR US
24         ;
25         ; Set up info about data to be moved
26         ;
27         000660 016703 000004G      MOV     EMTBLK+4,R3      ;GET USER'S BUFFER ADDRESS
28         000664 016705 000006G      MOV     EMTBLK+6,R5      ;GET # WORDS TO MOVE
29         000670 010567 000000G      MOV     R5,UR0          ;RETURN WORD COUNT TO USER IN R0
30         000674 001523             BEQ      SDMVX          ;BR IF NOTHING TO WRITE
31         000676 006305             ASL      R5              ;CONVERT TO BYTE COUNT
32         000700 060305             ADD     R3,R5             ;GET ADDRESS PAST END OF USER'S BUFFER
33         ;
34         ; See if this is the first write to the spooled device.
35         ;
36         000702 132761 000000G 000000G      BITB     #SF$1ST,SFFLAG(R1); IS THIS 1ST WRITE?
37         000710 001073             BNE      SDMV1          ;BR IF NOT 1ST WRITE
38         ;
39         ; This is first write to this spooled device.
40         ; See if 1st write is to block # 0 or non-zero.
41         ;
42         000712 005767 000002G      TST      EMTBLK+2          ;Is 1st block # 0?
43         000716 001403             BEQ      2$              ;Br if yes
44         000720 152761 000000G 000000G      BISB     #SF$BN1,SFFLAG(R1);Remember 1st block is not 0
45         ;
46         ; See if user specified a form name.
47         ;
48         000726 152761 000000G 000000G 2$:    BISB     #SF$1ST,SFFLAG(R1);REMEMBER 1ST WRITE HAS BEEN DONE
49         000734 004767 000000G      CALL     GETUCH          ;GET CHAR FROM USER'S BUFFER
50         000740 005303             DEC      R3              ;POINT BACK TO CHAR JUST GOTTEN
51         000742 120027 000135      CMPB     R0,#RSQBR        ;DID USER SPECIFY A FORM NAME?
52         000746 001037             BNE      16$            ;BR IF NOT
53         ;
54         ; User specified a form name with this file.
55         ; Move the name to buffer in SFCB.
56         ;
57         000750 010102             MOV     R1,R2             ;GET ADDRESS OF SFCB

```

SDMOVE -- Write user's data to spool file

```

58 000752 062702 000000G      ADD    #SFFORM,R2      ;POINT TO FORM NAME BUFFER IN SFCB
59 000756 005203              INC    R3              ;SKIP OVER RIGHT-SQUARE-BRACKET CHAR
60 000760 012704 000000G      MOV    #6,R4          ;FORM NAMES MAY BE UP TO 6 CHARS LONG
61 000764 004767 000000G      13$:  CALL   GETUCH      ;GET CHAR FROM USER'S BUFFER
62 000770 120027 000015      CMPB   R0,#CR          ;CARRIAGE-RETURN?
63 000774 001412              BEQ    14$            ;BR IF HIT END OF NAME
64 000776 120027 000141      CMPB   R0,#141         ;Is this a lower case character?
65 001002 103405              BLO    1$            ;Br if not
66 001004 120027 000172      CMPB   R0,#172
67 001010 101002              BHI    1$
68 001012 162700 000040      SUB     #40,R0          ;Convert lower case char to upper case
69 001016 110022              1$:  MOVB   R0,(R2)+      ;MOVE NAME TO SFCB BUFFER
70 001020 077417              SOB     R4,13$          ;MOVE UP TO 6 CHARS
71 001022 004767 000000G      14$:  CALL   GETUCH      ;NOW SKIP UP LINE-FEED
72 001026 120027 000012      CMPB   R0,#LF
73 001032 001373              BNE    14$
74 001034 005704              TST     R4              ;DID WE GET ALL 6 CHARACTERS?
75 001036 001420              BEQ     SDMV1          ;BR IF YES
76 001040 112722 000040      15$:  MOVB   #'',(R2)+      ;PAD WITH BLANKS
77 001044 077403              SOB     R4,15$
78 001046 132761 000000G 000000G 16$:  BITB   #SF$BN1,SFFLAG(R1);Is 1st block is not 0?
79 001054 001011              BNE    SDMV1          ;NO FLAGPAGE IF NOT
80 001056 016100 000000G      MOV     SFSDCB(R1),R0    ;GET SDCB FOR FILE
81 001062 032760 000000G 000000G      BIT     #SD$FLG,SDFLAG(R0);FLAG PAGE FOR DEVICE?
82 001070 001403              BEQ     SDMV1          ;BR IF NOT
83 001072              OCALL   FLGPAG          ;ELSE DO FLAG PAGE
84
85      ; Move data from user's buffer to spool buffer in job context area.
86
87 001100 016702 000000G      SDMV1:  MOV     SXBPNT,R2      ;GET CURRENT POINTER INTO SPOOL BUFFER
88 001104 020227 000000G      2$:  CMP     R2,#SPUBND      ;HAVE WE REACHED END OF SPOOL BUFFER?
89 001110 103406              BLO    1$            ;BR IF NOT
90 001112 010267 000000G      MOV     R2,SXBPNT        ;SAVE BUFFER POINTER
91 001116 004767 000036      CALL   SBUFOT          ;WRITE BUFFER TO SPOOL FILE
92 001122 016702 000000G      MOV     SXBPNT,R2        ;GET POINTER INTO SPOOL BUFFER
93 001126 004767 000000G      1$:  CALL   GETUCH      ;GET CHARACTER FROM USER'S BUFFER
94 001132 110022              MOVB   R0,(R2)+      ;MOVE CHAR TO SPOOL BUFFER
95 001134 020305              CMP     R3,R5          ;HAVE WE MOVED ALL OF USER'S DATA?
96 001136 103762              BLO    2$            ;BR IF NOT
97
98      ; Finished moving user's data
99
100 001140 010267 000000G      MOV     R2,SXBPNT        ;SAVE SPOOL BUFFER POINTER
101
102      ; Finished
103
104 001144 012605      SDMVX:  MOV     (SP)+,R5
105 001146 012604      MOV     (SP)+,R4
106 001150 012603      MOV     (SP)+,R3
107 001152 012602      MOV     (SP)+,R2
108 001154 012601      MOV     (SP)+,R1
109 001156 000207      RETURN

```

1				.SBTTL SBUFOT -- Write current buffer to file
2				-----
3				; SBUFOT IS CALLED TO WRITE THE CURRENT SPOOL BUFFER TO
4				; THE SPOOL FILE AND THEN RESET THE BUFFER POINTERS AND
5				; ALLOCATE A NEW DISK BLOCK FOR THE NEXT BUFFER.
6				; WHEN CALLED, SXSCFB MUST POINT TO THE SFCB FOR THE
7				; CURRENT FILE. ALL REGISTERS ARE PRESERVED.
8				; Inputs:
9				; CHNNUM = Current channel number.
10				; SPUBUF = Spool buffer.
11				; SXBPNT = Pointer to free space in spool buffer.
12				;
13				; Outputs:
14				; SXBPNT = Reset to point to first free byte in buffer.
15				;
16	001160	010046		SBUFOT: MOV RO, -(SP)
17	001162	010146		MOV R1, -(SP)
18	001164	016701	0000000	MOV SXSCFB, R1 ; POINT TO SFCB
19	001170	001415		BEG 9% ; BRANCH IF NO FILE USING BUFFER
20				; CHECK NUMBER OF DATA WORDS IN BUFFER
21	001172	016700	0000000	MOV SXBPNT, RO ; GET CURRENT POINTER
22	001176	162700	0000000	SUB #SPUBUF, RO ; SUBTRACT BUFFER BASE
23	001202	162700	0000004	SUB #4, RO ; REMOVE CONTROL WORDS
24	001206	001406		BEG 9% ; BRANCH IF BUFFER IS EMPTY
25				; GET A NEW DISK BLOCK AND SET AS FILE FLINK IN THIS BUFFER
26	001210	004767	003070	CALL GTSBLK ; GET A DISK BLOCK
27	001214	010067	0000000	MOV RO, SPUBUF ; SET AS FLINK IN BUFFER
28				; WRITE OUT THE BUFFER.
29	001220	004767	000042	CALL SFLUSH ; DO THE WRITE
30				;
31	001224	012601		9%: MOV (SP)+, R1
32	001226	012600		MOV (SP)+, RO
33	001230	000207		RETURN

SGETBF -- Reserve buffer for file

```

1          .SBTTL  SGETBF -- Reserve buffer for file
2          -----
3          ; SGETBF IS CALLED TO MAKE SURE THAT THE SPOOL BUFFER IS
4          ; IN USE BY THE SPOOL FILE WHOSE SFCB IS POINTED TO BY R1.
5          ; IF THE BUFFER IS BEING USED BY ANOTHER FILE, THE BUFFER
6          ; IS WRITTEN OUT WITH ITS CURRENT CONTENTS AND
7          ; THE BUFFER IS CLAIMED FOR THE NEW FILE BY STORING
8          ; THE SFCB ADDRESS IN SXSFCB.
9          ; ALL REGISTERS ARE PRESERVED.
10         ;
11 001232 020167 000000G SGETBF: CMP      R1,SXSFCB      ; HAVE WE GOT CONTROL OF BUFFER?
12 001236 001412          BEQ      9$              ; BRANCH IF YES
13 001240 005767 000000G          TST      SXSFCB      ; IS BUFFER IN USE BY ANYONE?
14 001244 001402          BEQ      8$              ; BRANCH IF NOT
15          ; BUFFER IS IN USE BY ANOTHER FILE.
16          ; WRITE OUT INFO IN BUFFER.
17 001246 004767 177706          CALL     SBUFOT      ; DUMP CURRENT BUFFER FULL
18          ; NOW CLAIM BUFFER FOR OUR FILE
19 001252 010167 000000G 8$:      MOV      R1,SXSFCB      ; SAY WE'VE GOT BUFFER
20 001256 012767 000004G 000000G      MOV      #SPUBUF+4, SXBPNT; INITIALIZE SPOOL BUFFER POINTER
21 001264 000207          9$:      RETURN

```

SFLUSH -- Write current buffer to file

```

1          .SBTTL  SFLUSH -- Write current buffer to file
2          ;-----
3          ; SFLUSH IS CALLED TO WRITE THE CURRENT SPOOL BUFFER TO
4          ; THE SPOOL FILE.  SFLUSH SETS THE CORRECT WORD COUNT
5          ; IN THE BUFFER CONTROL WORD AND WRITES THE BUFFER TO
6          ; THE DISK BLOCK WHOSE ADDRESS IS IN SFFLNK.
7          ; IT THEN SETS SFFLNK TO THE VALUE WHICH IS IN THE
8          ; FLINK FIELD OF THE BUFFER.  THE BUFFER POINTERS
9          ; ARE RESET.  ALL REGISTERS ARE PRESERVED.
10         ;
11 001266 010046 SFLUSH: MOV      R0,-(SP)
12 001270 010146      MOV      R1,-(SP)
13 001272 010246      MOV      R2,-(SP)
14 001274 010346      MOV      R3,-(SP)
15 001276 012702 000000G      MOV      #SPUBUF,R2      ; POINT TO BUFFER
16 001302 016701 000000G      MOV      SXSFCB,R1      ; POINT TO CURRENT SFCB
17 001306 001456      BEQ      8$      ; BRANCH IF NO FILE USING BUFFER
18         ; SET WORD COUNT IN BUFFER
19 001310 016700 000000G      MOV      SXBPNT,R0      ; GET CURRENT POINTER INTO BUFFER
20 001314 032700 000001      BIT      #1,R0      ; IS ADDRESS ODD?
21 001320 001401      BEQ      1$      ; BRANCH IF NOT
22 001322 105020      CLRB      (R0)+      ; PUT NULL AS ODD CHAR
23 001324 160200      1$: SUB      R2,R0      ; SUBTRACT BUFFER BASE
24 001326 162700 000004      SUB      #4,R0      ; DON'T COUNT CONTROL WORDS
25 001332 006200      ASR      R0      ; CONVERT COUNT TO WORDS
26 001334 010062 000002      MOV      R0,2(R2)      ; STORE WORD COUNT INTO BUFFER
27         ; WRITE OUT THE BUFFER.
28 001340      2$: .WRITW      #SPLARG,#USPLCH,R2,#256.,SFFLNK(R1) ; WRITE BLOCK TO SPOOL FILE
29 001376 103003      BCC      3$      ; BR IF NO WRITE ERROR OCCURED
30 001400      .PRINT      #SPLERR      ; SIGNIFY ERROR ON SPOOL WRITE
31 001406 011261 000000G      3$: MOV      (R2),SFFLNK(R1) ; SET NEW FILE FLINK
32         ; TELL SPOOLER THAT ANOTHER BLOCK IS AVAILABLE.
33 001412 005261 000000G      INC      SFNMBL(R1)      ; SAY ANOTHER BLOCK IS IN FILE
34 001416 016100 000000G      MOV      SFSDCB(R1),R0      ; POINT TO SDCB
35 001422 132761 000000G 000000G      BITB      #SF#BSY,SFFLAG(R1); IS FILE ACTIVE NOW?
36 001430 001403      BEQ      7$      ; BRANCH IF NOT
37 001432 004767 000712      CALL      SPOLRD      ; TELL SPOOLER IT CAN READ BLOCK
38 001436 000402      BR      8$
39 001440 004767 000100      7$: CALL      SPOLGO      ; TRY TO START SPOOLER
40         ; RESET BUFFER POINTERS
41 001444 062702 000004      8$: ADD      #4,R2      ; POINT PAST CONTROL WORDS
42 001450 010267 000000G      MOV      R2,SXBPNT
43 001454 012603      MOV      (SP)+,R3
44 001456 012602      MOV      (SP)+,R2
45 001460 012601      MOV      (SP)+,R1
46 001462 012600      MOV      (SP)+,R0
47 001464 000207      RETURN

```

SPLSHF -- Set or reset hold flag for file

```

1          .SBTTL  SPLSHF -- Set or reset hold flag for file
2          ;-----
3          ; SPLSHF is called to set or reset the hold flag for a spool file.
4          ;
5          ; Inputs:
6          ;   CHNNUM = User's channel number
7          ;   R2 = 0==>nohold; 1==>hold file.
8          ;
9 001466 010146 SPLSHF: MOV      R1, -(SP)
10 001470 012767 000000G 000000G      MOV      #SD$RSV, URO      ; Assume the device is not spooled
11          ;
12          ; Find SFCF associated with this channel
13          ;
14 001476 004767 003220      CALL      FNDSF      ; Locate SFCF associated with this channel
15 001502 103416      BCS      9$      ; Br if cannot find the SFCF
16          ;
17          ; Set or reset the flag in the SFCB
18          ;
19 001504 116167 000000G 000000G      MOV      SFFLAG(R1), URO      ; Get current setting of hold mode
20 001512 042767 000000C 000000G      BIC      #^CSF$HLD, URO      ; And return to user
21 001520 152761 000000G 000000G      BIS      #SF$HLD, SFFLAG(R1); Assume we want to set the flag
22 001526 005702      TST      R2      ; Do we want to reset the flag?
23 001530 001003      BNE      9$      ; Br if not
24 001532 142761 000000G 000000G      BIC      #SF$HLD, SFFLAG(R1); Clear the hold flag
25          ;
26          ; Finished
27          ;
28 001540 012601 9$:      MOV      (SP)+, R1
29 001542 000207      RETURN

```

```

1          .SBTTL  SPOLGO -- Try to start spooled device
2          ;-----
3          ; SPOLGO IS CALLED TO TRY TO START UP THE SPOOLED DEVICE
4          ; WHOSE SDCB IS POINTED TO BY RO.
5          ; ALL REGISTERS ARE PRESERVED.
6          ;
7 001544    SPOLGO: DISABL          ;** DISABLE **
8 001552    TST      SDSFCB(RO)      ;IS DEVICE BUSY NOW?
9 001556    BEQ      7$              ;BR IF NOT
10 001560    JMP      SGX2
11 001564    010146    7$:  MOV      R1,-(SP)
12          ;
13          ; LOOK FOR WAITING FORM ALIGNMENT PRINT FILE.
14          ; (ALIGNMENT FILES USE THE FORM NAME "*** ")
15          ;
16 001566    016001    000000G      MOV      SDFHD(RO),R1      ;POINT TO 1ST WAITING SFCB
17 001572    001002      BNE      1$              ;BR IF THERE ARE WAITING FILES
18 001574    000167    000536      JMP      SGX1
19 001600    005761    000000G      1$:  TST      SFNMBL(R1)      ;ANY DATA IN FILE YET?
20 001604    001404      BEQ      5$              ;BR IF NOT
21 001606    026127    000000G 025052  CMP      SFFORM(R1),#"**" ;ALIGNMENT FILE?
22 001614    001576      BEQ      SGOTFL          ;BR IF FOUND ALIGNMENT FILE
23 001616    016101    000000G      5$:  MOV      SFQLNK(R1),R1    ;LINK FORWARD TO NEXT SFCB
24 001622    001366      BNE      1$              ;BR IF MORE TO CHECK
25          ;
26          ; SEE IF WE ARE WAITING FOR A FORM MOUNT.
27          ;
28 001624    032760    000000G 000000G  BIT      #SD$WFM,SDFLAG(RO);WAITING FOR A FORM MOUNT?
29 001632    001402      BEQ      9$              ;BR IF NOT
30 001634    000167    000476      JMP      SGX1          ;EXIT IF WAITING FOR FORM MOUNT
31          ;
32          ; SEE IF WE CAN FIND A FILE WHICH NEEDS THE FORM WHICH
33          ; IS CURRENTLY MOUNTED.
34          ;
35 001640    016001    000000G      9$:  MOV      SDFHD(RO),R1      ;POINT TO 1ST WAITING SFCB
36 001644    132761    000000G 000000G  3$:  BITB     #SF$HLD,SFFLAG(R1);IS THIS FILE BEING HELD?
37 001652    001031      BNE      2$              ;BR IF YES
38 001654    005761    000000G      TST      SFNMBL(R1)      ;ANY DATA IN FILE YET?
39 001660    001426      BEQ      2$              ;BR IF NOT
40 001662    132761    000000G 000000G  BITB     #SF$1ST,SFFLAG(R1);WAS 1ST WRITE EVER DONE TO FILE
41 001670    001002      BNE      8$              ;BR IF YES
42 001672    000167    000314      JMP      SGOTFL          ;IF NOT THEN IGNORE FORM NAME (EMPTY FILE)
43 001676    132761    000000G 000000G  8$:  BITB     #SF$DEL,SFFLAG(R1);Is this file to be deleted?
44 001704    001142      BNE      SGOTFL          ;Br if yes -- Delete the file now
45 001706    026160    000000G 000000G  CMP      SFFORM(R1),SDFORM(RO);SEE IF SAME FORM NAME
46 001714    001010      BNE      2$
47 001716    026160    000002G 000002G  CMP      SFFORM+2(R1),SDFORM+2(RO)
48 001724    001004      BNE      2$
49 001726    026160    000004G 000004G  CMP      SFFORM+4(R1),SDFORM+4(RO)
50 001734    001424      BEQ      SGFSF          ;BR IF FOUND ELIGIBLE FILE
51 001736    016101    000000G      2$:  MOV      SFQLNK(R1),R1    ;LINK TO NEXT SFCB
52 001742    001340      BNE      3$              ;BR IF MORE TO CHECK
53          ;
54          ; CAN'T FIND FILE NEEDING CURRENTLY MOUNTED FORM.
55          ; SEE IF WE ARE ALLOWED TO REQUEST NEW FORM.
56          ;
57 001744    032760    000000G 000000G  BIT      #SD$FLK,SDFLAG(RO);IS CURRENT FORM LOCKED ON PRINTER?

```

```

58 001752 001171          BNE      SGX1          ;BR IF FORM LOCKED
59
60          ; WE CAN REQUEST FORM CHANGE.
61          ; LOOK FOR 1ST FILE WAITING TO BE PRINTED.
62
63 001754 016001 000000G      MOV      SDFHD(R0),R1      ;POINT TO 1ST SFCB
64 001760 005761 000000G      4$:      TST      SFNMBL(R1)      ;ANY DATA IN FILE YET?
65 001764 001404              BEQ      6$              ;BR IF NOT
66 001766 132761 000000G 000000G      BITB     #SF$HLD,SFFLAG(R1); IS FILE BEING HELD?
67 001774 001414              BEQ      SGNFMT           ;BR IF GOT ELIGIBLE FILE
68 001776 016101 000000G      6$:      MOV      SFQLNK(R1),R1      ;LINK FORWARD TO NEXT SFCB
69 002002 001366              BNE      4$              ;BR IF MORE TO CHECK
70 002004 000554              BR       SGX1           ;COULD FIND NO FILE TO PRINT
71
72          ; FOUND FILE FOR CURRENTLY MOUNTED FORM.
73          ; SEE IF IN SINGLE FILE MODE WHICH REQUIRES A MOUNT MESSAGE
74          ; EVEN FOR THIS FILE.
75
76 002006 032760 000000G 000000G      SGFSF:  BIT      #SD$SNG,SDFLAG(R0); IN SINGLE FILE MODE?
77 002014 001476              BEQ      SGOTFL           ;BR IF NOT
78 002016 032760 000000G 000000G      BIT      #SD$SMS,SDFLAG(R0); ALREADY SENT MESSAGE?
79 002024 001072              BNE      SGOTFL           ;BR IF YES
80
81          ; PUT OUT MESSAGE REQUESTING FORM MOUNT.
82
83 002026              SGNFMT:  ENABL              ;** ENABLE **
84 002034 010246              MOV      R2,-(SP)
85 002036 010446              MOV      R4,-(SP)
86 002040 010546              MOV      R5,-(SP)
87 002042 010046              MOV      R0,-(SP)      ;SAVE SDCB ADDRESS ON TOP OF STACK
88 002044 116702 000000G      MOVB     CTRLTT,R2      ;GET JOB # OF OPERATOR'S CONSOLE
89 002050 001445              BEQ      2$              ;BR IF NO OPERATOR
90 002052 032762 000000G 000000G      BIT      #DILUP,LSW(R2) ; IS OPERATOR LOGGED ON NOW?
91 002060 001441              BEQ      2$              ;BR IF NOT
92 002062 004767 000000G      CALL     GTSYMB          ;GET A FREE SYSTEM MESSAGE BUFFER
93 002066 103436              BCS      2$              ;BR IF NONE AVAILABLE
94 002070 010402              MOV      R4,R2          ;GET ADDRESS OF MESSAGE BUFFER
95 002072 062702 000000G      ADD      #SB$TXT,R2      ;POINT TO TEXT STORAGE AREA
96 002076 012705 000000G      MOV      #FMMSG,R5      ;MOVE MESSAGE TEXT TO BUFFER
97 002102 012700 000021      MOV      #17,R0          ;MOVE 17 WORDS
98 002106 012522              1$:      MOV      (R5)+,(R2)+
99 002110 077002              SOB      R0,1$
100 002112 010402              MOV      R4,R2          ;GET ADDRESS OF MESSAGE BUFFER BLOCK
101 002114 062702 000014G      ADD      #SB$TXT+12.,R2 ;POINT INTO MESSAGE TEXT
102 002120 062701 000000G      ADD      #SFFORM,R1      ;MOVE FORM NAME INTO MESSAGE TEXT
103 002124 012122              MOV      (R1)+,(R2)+
104 002126 012122              MOV      (R1)+,(R2)+
105 002130 012122              MOV      (R1)+,(R2)+
106 002132 011600              MOV      (SP),R0        ;GET SDCB ADDRESS
107 002134 062700 000000G      ADD      #SDANAM,R0      ;Point to ASCII device name
108 002140 062702 000012      ADD      #10.,R2         ;Point to position where dev name goes
109 002144 112022              MOVB     (R0)+,(R2)+    ;Move device name into message
110 002146 112022              MOVB     (R0)+,(R2)+
111 002150 112022              MOVB     (R0)+,(R2)+
112 002152 116701 000000G      MOVB     CTRLTT,R1      ;GET # OF OPERATOR'S CONSOLE
113 002156              OCALL     QMSG              ;QUEUE MESSAGE FOR OPERATOR
114 002164 012600              2$:      MOV      (SP)+,R0

```

```

115 002166 052760 000000G 000000G      BIS      #SD$WFM,SDFLAG(R0); WAITING FOR FORM MOUNT
116 002174 052760 000000G 000000G      BIS      #SD$SMS,SDFLAG(R0); FORM MOUNT MESSAGE SENT
117 002202 012605                MOV      (SP)+,R5
118 002204 012604                MOV      (SP)+,R4
119 002206 012602                MOV      (SP)+,R2
120 002210 000452                BR       SGX1
121
122          ;      FOUND ELIGIBLE FILE -- SAY DEVICE IS BUSY WITH FILE.
123
124 002212 010160 000000G      SGOTFL: MOV      R1,SDSFCB(R0)      ; ASSOCIATE FILE WITH DEVICE
125 002216 152761 000000G 000000G      BISB     #SF$BSY,SFFLAG(R1); SAY FILE IS BUSY
126 002224                ENABL                      ; ** ENABLE **
127 002232 010146                MOV      R1,-(SP)
128 002234 012701 000000G      MOV      #DF$ENT,R1      ; Get function code for .ENTER
129 002240 004767 001304      CALL     SDFUNC      ; Tell device handler we are doing .ENTER
130 002244 012601                MOV      (SP)+,R1
131 002246 132761 000000G 000000G      BITB     #SF$DEL,SFFLAG(R1); Is file marked to be deleted?
132 002254 001403                BEQ      1$      ; Br if not
133 002256 052760 000000G 000000G      BIS      #SD$DEL,SDFLAG(R0); Set delete flag for spooler
134 002264 016160 000000G 000000G 1$: MOV      SFSTRT(R1),SDFLNK(R0); SET DISK ADDR FOR 1ST READ
135 002272 005060 000000G      CLR      SDBLK(R0)      ; Say 1st block written will be # 0
136 002276 132761 000000G 000000G      BITB     #SF$BN1,SFFLAG(R1); Should 1st block written be # 1?
137 002304 001402                BEQ      2$      ; Br if not
138 002306 005260 000000G      INC      SDBLK(R0)      ; Make 1st block written be # 1
139 002312 005367 000000G 2$: DEC      NESB      ; ONE LESS EXTRA SPOOL BUFFER
140 002316 004767 000026      CALL     SPOLRD      ; READ IN 1ST BLOCK FROM FILE
141          ;      RESTART USERS WAITING FOR DISK BLOCKS
142 002322 010046                MOV      RO,-(SP)
143 002324 012700 000000G      MOV      #S$SPDB,RO      ; POINT TO WAITING Q HEAD
144 002330 004767 000000G      CALL     UREGO      ; RESTART 1ST USER IN Q
145 002334 012600                MOV      (SP)+,RO
146 002336 012601      SGX1: MOV      (SP)+,R1
147 002340      SGX2: ENABL                      ; ** ENABLE **
148 002346 000207                RETURN

```

```

1          .SBTTL  SPOLRD -- Read next block from spool file
2          ;-----
3          ; SPOLRD IS CALLED TO ATTEMPT TO READ THE NEXT BLOCK
4          ; FROM THE SPOOL FILE.  WHEN CALLED, R0 MUST POINT TO THE
5          ; SDCB FOR THE DEVICE ATTACHED TO THE FILE.
6          ; ALL REGISTERS ARE PRESERVED.
7          ;
8 002350 010146 SPOLRD: MOV      R1,-(SP)
9 002352 010246      MOV      R2,-(SP)
10 002354 016001 000000G      MOV      SDSFCB(R0),R1  ;POINT TO SFCB
11 002360 001534      BEQ      SRX          ;BRANCH IF DEVICE INACTIVE
12          ; SEE IF WE SHOULD START ANOTHER READ.
13 002362      DISABL          ;** DISABLE **
14 002370 005760 000000G      TST      SDBUF2(R0)      ;INPUT BUFFER ALREADY BUSY?
15 002374 001123      BNE      SREX          ;BR IF YES
16 002376 032760 000000G 000000G      BIT      #SD$BWT,SDFLAG(R0) ;ALREADY WAITING FOR A BUFFER?
17 002404 001117      BNE      SREX          ;BR IF YES
18 002406 032760 000000G 000000G      BIT      #SD$BAK,SDFLAG(R0); IS BACKUP WANTED?
19 002414 001006      BNE      3$          ;BR IF YES
20 002416 005761 000000G      TST      SFNMBL(R1)      ;ANY BLOCKS IN THIS FILE?
21 002422 001510      BEQ      SREX          ;BR IF FILE IS EMPTY
22 002424 005760 000000G      TST      SDFLNK(R0)      ;DO WE HAVE DISK FLINK?
23 002430 001505      BEQ      SREX          ;BR IF NOT
24 002432 005760 000000G 3$: TST      SDBUF1(R0)      ;OUTPUT BUFFER ALLOCATED?
25 002436 001403      BEQ      2$          ;IF NOT DEFINITELY START READ
26          ; SEE IF THERE ARE ENOUGH BUFFERS AVAILABLE FOR US TO
27          ; DOUBLE BUFFER.  ONLY DOUBLE BUFFER IF THERE ARE ENOUGH FREE
28          ; BUFFERS FOR EACH ACTIVE SPOOLER TO GET AT LEAST ONE.
29 002440 005727 000000C      TST      #<SPLNB+1-<2*SPLND>>; ENOUGH BUFFERS TO DOUBLE BUFFER?
30 002444 002477      BLT      SREX          ;BR IF TOO FEW BUFFERS TO DOUBLE BUFFER
31          ; WE WANT TO START THE NEXT READ.
32          ; SEE IF WE CAN GET A SPOOL BUFFER.
33 002446 016702 000000G 2$: MOV      SPLBHD,R2          ;HEAD OF FREE BUFFER LIST
34 002452 001010      BNE      1$          ;BRANCH IF BUFFER AVAILABLE
35          ; THERE ARE NO FREE BUFFERS.
36          ; QUEUE SPOLRD TO BE RECALLED WHEN A BUFFER IS RELEASED.
37 002454 052760 000000G 000000G      BIS      #SD$BWT,SDFLAG(R0) ;SAY WE ARE WAITING ON A BUFFER
38 002462 012701 000006'      MOV      #SPBWH, R1          ;POINT TO WAIT LIST HEAD
39 002466 004767 000166      CALL      SPOOLQ          ;QUEUE FOR RECALL ** ENABLE **
40 002472 000467      BR      SRX
41          ; WE FOUND A FREE BUFFER -- START READ INTO IT
42 002474 011267 000000G 1$: MOV      (R2),SPLBHD      ;REMOVE BUFFER FROM FREE LIST
43 002500 010260 000000G      MOV      R2,SDBUF2(R0)    ;SET BUFFER ADDRESS
44 002504 052760 000000G 000000G      BIS      #SD$INR,SDFLAG(R0) ;SAY READ IS IN PROGRESS
45 002512      ENABL          ;** ENABLE **
46          ;
47          ; SEE IF OPERATOR WANTS TO BACK UP IN SPOOL FILE
48          ;
49 002520 032760 000000G 000000G      BIT      #SD$BAK,SDFLAG(R0); HAS BACKUP BEEN REQUESTED?
50 002526 001417      BEQ      4$          ;BR IF NOT
51          ; BACK UP HAS BEEN REQUESTED.  BACK UP TO OLDEST REMEMBERED BLOCK.
52 002530 010346      MOV      R3,-(SP)
53 002532 010003      MOV      R0,R3          ;POINT TO CELLS WITH SAVED
54 002534 062703 000000G      ADD      #SDBU,R3          ;BACK-UP DISK ADDRESSES
55 002540 011360 000000G      MOV      (R3),SDFLNK(R0) ;SET OLDEST AS NEW FLINK
56 002544 005261 000000G 5$: INC      SFNMBL(R1)      ;WE JUST ADDED 1 BLOCK TO FILE
57 002550 005023      CLR      (R3)+          ;CLEAR SAVE AREA

```

SPOLRD -- Read next block from spool file

```

58 002552 005713          TST      (R3)          ;MORE TO ADD?
59 002554 001373          BNE      5$            ;ADD ALL WE REMEMBER
60 002556 042760 000000G 000000G          BIC      #SD$BAK,SDFLAG(R0);SAY BACKUP HAS BEEN DONE
61 002564 012603          MOV      (SP)+,R3
62
63          ; START THE DISK READ
64
65 002566 010046          4$:      MOV      R0,-(SP)      ;SAVE ADDRESS OF SDCB
66
67          ; Get a system I/O queue element
68
69 002570 012701 000000G          MOV      #SPLCHN,R1      ;POINT TO SPOOL FILE CHANNEL BLOCK
70 002574 004767 000000G          CALL     GETSYQ        ;GET SYSTEM I/O QUEUE ELEMENT
71
72          ; Set up I/O queue element for read
73
74 002600 011600          MOV      (SP),R0      ;GET ADDRESS OF SDCB
75 002602 010061 000000G          MOV      R0,Q.CHAN(R1)  ;SDCB ADDRESS WILL BE IN R1 FOR COMPLETION ROUTINE
76 002606 066061 000000G 000000G          ADD      SDFLNK(R0),Q.BLKN(R1);SET DISK BLOCK #
77 002614 010200          MOV      R2,R0      ;GET BUFFER ADDRESS
78 002616 004767 000000G          CALL     SYBFAD        ;SET BUFFER ADDRESS IN I/O QUEUE ELEMENT
79 002622 012761 000000G 000000G          MOV      #SBUFWD,Q.WCNT(R1);SET WORD COUNT FOR TRANSFER
80 002630 012761 002734' 000000G          MOV      #SFRCMP,Q.COMP(R1);SET ADDRESS OF COMPLETION ROUTINE
81 002636 012600          MOV      (SP)+,R0
82
83          ; Start the I/O
84
85 002640 004767 000000G          CALL     SYQIO        ;QUEUE THE I/O REQUEST
86
87          ; Finished
88
89 002644          SREX:  ENABL                      ;*** ENABLE ***
90 002652 012602          SRX:  MOV      (SP)+,R2
91 002654 012601          MOV      (SP)+,R1
92 002656 000207          RETURN

```

1			.SBTTL SPOOLQ -- Put spooler in wait queue
2			-----
3			; SPOOLQ IS CALLED TO ADD A SPOOLED DEVICE CONTROL BLOCK
4			; TO THE END OF SOME QUEUE WHOSE LIST HEAD IS POINTED
5			; TO BY R1. WHEN CALLED R0 MUST CONTAIN THE ADDRESS
6			; OF THE SDCB.
7			; R1 IS DESTROYED -- ALL OTHER REGISTERS ARE PRESERVED.
8			;
9	002660	010246	SPOOLQ: MOV R2,-(SP)
10	002662		DISABL ;** DISABLE **
11	002670	011102	MOV (R1),R2 ;GET CONTENTS OF LIST HEAD
12	002672	001410	BEG 1\$;BRANCH IF LIST IS EMPTY
13			; LIST NOT EMPTY -- PUT US AT TAIL
14	002674	016201 000000G	3\$: MOV SDWLST(R2),R1 ;LOOK FOR END OF LIST
15	002700	001402	BEG 2\$
16	002702	010102	MOV R1,R2 ;FOLLOW CHAIN
17	002704	000773	BR 3\$
18	002706	010062 000000G	2\$: MOV R0,SDWLST(R2) ;ADD US TO END OF LIST
19	002712	000401	BR 4\$
20			; EMPTY LIST -- PUT US AT HEAD.
21	002714	010011	1\$: MOV R0,(R1) ;STORE INTO LIST HEAD
22	002716	005060 000000G	4\$: CLR SDWLST(R0) ;SAY WE ARE TAIL OF LIST
23	002722		ENABL ;** ENABLE **
24	002730	012602	MOV (SP)+,R2
25	002732	000207	RETURN

```

1          .SBTTL  SFRCMP -- Spool read completion routine
2          ;-----
3          ; SPOOL READ COMPLETION ROUTINE -- ENTERED WHEN A READ
4          ; FROM THE SPOOL FILE FINISHES.
5          ;
6          ; Inputs:
7          ; R1 = Address of SDCB.
8          ;
9          ;
10         ; ADD THE DISK ADDRESS OF THE BLOCK JUST READ TO THE END
11         ; OF THE LIST OF BLOCKS WHICH ARE BEING REMEMBERED IN CASE
12         ; THE OPERATOR WANTS TO BACK UP IN THE SPOOL FILE.
13         ; SEE IF THERE IS ROOM LEFT FOR ANOTHER BLOCK ADDRESS.
14         ;
15 002734 010100 SFRCMP: MOV R1,R0 ; MOVE SDCB ADDRESS TO R0
16 002736 005760 000000G TST SDBULS(R0) ; IS TABLE FULL?
17 002742 001412 BEQ 1$ ; BR IF TABLE NOT FULL
18         ; TABLE IS FULL. FORGET OLDEST ENTRY.
19 002744 016001 000000G MOV SDBU(R0),R1 ; GET OLDEST DISK ADDRESS IN TABLE
20 002750 004767 001544 CALL RLSBLK ; RELEASE THE BLOCK
21         ; SHOVE UP ALL TABLE ENTRIES
22 002754 010001 MOV R0,R1 ; POINT INTO TABLE
23 002756 062701 000002G ADD #SDBU+2,R1
24 002762 012161 177774 2$: MOV (R1)+,-4(R1) ; SHIFT UP OVER 1ST ENTRY
25 002766 001375 BNE 2$ ; LOOP UNTIL ALL MOVED
26         ; ADD NEW ENTRY TO 1ST FREE SLOT IN TABLE
27 002770 010001 1$: MOV R0,R1
28 002772 062701 000000G ADD #SDBU,R1
29 002776 005721 3$: TST (R1)+ ; IS THIS SLOT EMPTY?
30 003000 001376 BNE 3$ ; SEARCH FOR 1ST FREE SLOT
31 003002 016061 000000G 177776 MOV SDFLNK(R0),-2(R1) ; REMEMBER THIS DISK ADDRESS
32 003010 016001 000000G MOV SDSFCB(R0),R1 ; POINT TO SFCB
33 003014 005361 000000G DEC SFNMBL(R1) ; SAY 1 LESS BLOCK IN FILE
34 003020 017060 000000G 000000G MOV @SDBUF2(R0),SDFLNK(R0) ; SET NEW DISK FLINK
35 003026 042760 000000G 000000G BIC #SD$INR,SDFLAG(R0) ; SAY READ IS FINISHED
36         ; SEE IF A WRITE TO SPOOLED DEVICE IS IN PROGRESS.
37         ; IF NOT, START ONE.
38 003034 005760 000000G TST SDBUF1(R0) ; OUTGOING BUFFER SET UP?
39 003040 001002 BNE 9$ ; BRANCH IF YES
40 003042 004767 000002G CALL SPOLWR ; START WRITE
41 003046 000207 9$: RETURN

```

```

1          .SBTTL  SPOLWR -- Write to spooled device
2          ;-----
3          ; SPOLWR IS CALLED TO START A WRITE OPERATION TO A
4          ; SPOOLED DEVICE.  WHEN CALLED R0 MUST POINT TO
5          ; THE SDCB FOR THE DEVICE.  ALL REGISTERS ARE PRESERVED.
6          ;
7 003050 010146      SPOLWR: MOV      R1,-(SP)
8 003052 010246      MOV      R2,-(SP)
9 003054              DISABL              ;** DISABLE **
10 003062 005760 000000G    TST      SDBUF1(R0)      ;IS WRITE IN PROGRESS NOW?
11 003066 001102          BNE      8$              ;BRANCH IF YES
12 003070 016002 000000G    MOV      SDBUF2(R0),R2    ;IS A BUFFER READY TO GO?
13 003074 001466          BEQ      7$              ;BRANCH IF NOT
14 003076 032760 000000G 000000G    BIT      #SD$INR,SDFLAG(R0) ;IS INPUT READ STILL GOING?
15 003104 001073          BNE      8$              ;BRANCH IF YES
16          ; THERE IS A BUFFER READY TO GO.  START WRITE.
17 003106 010260 000000G    MOV      R2,SDBUF1(R0)    ;SET ADDR OF OUTPUT BUFFER
18 003112 005060 000000G    CLR      SDBUF2(R0)      ;SAY NO INPUT BUFFER
19 003116          ENABL              ;** ENABLE **
20 003124 032760 000000G 000000G    BIT      #SD$DEL,SDFLAG(R0) ;ARE WE DELETING THIS FILE?
21 003132 001055          BNE      1$              ;BRANCH IF YES
22 003134 005760 000000G    TST      SDSKIP(R0)      ;ARE WE SKIPPING FORWARD?
23 003140 001403          BEQ      3$              ;BR IF NOT
24 003142 005360 000000G    DEC      SDSKIP(R0)      ;DEC REMAINING SKIP COUNT
25 003146 000447          BR       1$              ;SKIP THIS BLOCK
26 003150 005762 000002      3$: TST      2(R2)      ;IS WORD COUNT IN BUFFER 0?
27 003154 001444          BEQ      1$              ;BRANCH IF ZERO
28          ;
29          ; Write this block to the spooled device.
30          ;
31 003156 010046      2$: MOV      R0,-(SP)          ;SAVE SDCB ADDRESS
32          ;
33          ; Get a system I/O queue entry.
34          ;
35 003160 010001          MOV      R0,R1              ;POINT TO SDCB
36 003162 062701 000000G    ADD      #SD$CHAN,R1      ;POINT TO SPOOLED DEVICE CHANNEL BLOCK IN SDCB
37 003166 004767 000000G    CALL     GETSYQ          ;GET SYSTEM I/O QUEUE ELEMENT
38          ;
39          ; Set up I/O queue element for the write
40          ;
41 003172 011600          MOV      (SP),R0            ;GET ADDRESS OF SDCB
42 003174 066061 000000G 000000G    ADD      SDBLK(R0),Q.BLKN(R1) ;SET BLOCK NUMBER
43 003202 005260 000000G    INC      SDBLK(R0)        ;ADVANCE BLOCK NUMBER
44 003206 010200          MOV      R2,R0              ;GET BUFFER ADDRESS
45 003210 062700 000004      ADD      #4,R0          ;SKIP OVER BUFFER HEADER
46 003214 004767 000000G    CALL     SYBFAD          ;SET BUFFER ADDRESS IN I/O QUEUE ELEMENT
47 003220 016200 000002      MOV      2(R2),R0        ;GET # WORDS TO WRITE
48 003224 005400          NEG      R0                 ;NEGATIVE ==> WRITE OPERATION
49 003226 010061 000000G    MOV      R0,Q.WCNT(R1)    ;
50 003232 012761 003310' 000000G    MOV      #SWCMP,Q.COMP(R1) ;SET ADDRESS OF COMPLETION ROUTINE
51 003240 011661 000000G    MOV      (SP),Q.CHAN(R1) ;PUT SDCB ADDRESS IN R1 FOR COMPL ROUTINE
52          ;
53          ; Start the write
54          ;
55 003244 004767 000000G    CALL     SYQID            ;QUEUE THE I/O REQUEST
56 003250 012600          MOV      (SP)+,R0          ;GET BACK SDCB ADDRESS
57          ; TRY TO START READ OF NEXT BLOCK FROM FILE.

```

```

58 003252                                7$:      ENABL                                ; ** ENABLE **
59 003260    004767    177064              CALL      SPOLRD                                ; TRY TO DO READ
60 003264    000406                                BR      9$
61                                ; WE ARE DELETING THIS FILE SO IGNORE THIS BLOCK AND READ NEXT
62                                ; FAKE CALL TO WRITE COMPLETION ROUTINE.
63 003266    010001                                1$:      MOV      RO,R1                                ; GET SDCB ADDRESS
64 003270    004767    000014              CALL      SWCMP                                ; CALL WRITE COMPL ROUTINE
65 003274                                8$:      ENABL                                ; ** ENABLE **
66 003302    012602                                9$:      MOV      (SP)+,R2
67 003304    012601                                MOV      (SP)+,R1
68 003306    000207                                RETURN

```

```

1          .SBTTL  SWCMP  -- Spool write completion routine
2          ;-----
3          ; SWCMP IS THE COMPLETION ROUTINE WHICH IS ENTERED
4          ; ON COMPLETION OF A WRITE TO A SPOOLED DEVICE.
5          ; R1 IS DESTROYED (ALLOWED FOR COMPL ROUTINES).
6          ; ALL OTHER REGISTERS ARE PRESERVED.
7          ; Inputs:
8          ;   R1 = Address of SDCB
9          ;
10         SWCMP:  MOV     R1,R0          ;MOVE SDCB ADDRESS TO R0
11                 MOV     SDBUF1(R0),R1 ;POINT TO SPOOL BUFFER
12                 MOV     (R1),-(SP)    ;SAVE DISK FLINK FROM BUFFER
13                 CALL    RLSBUF        ;RELEASE SPOOL BUFFER
14                 CLR     SDBUF1(R0)    ;SAY WRITE IS FINISHED
15                 TST     (SP)+         ;IS FILE FLINK ZERO?
16                 BEQ     5$            ;BRANCH IF ZERO -- END OF FILE
17         ; THIS IS NOT THE END OF THE FILE.
18         ; TRY TO START NEXT WRITE.
19         CALL    SPOLWR                ;TRY FOR WRITE
20         RETURN
21         ; THIS IS THE END OF THE FILE.
22         5$:  BIT     #SD$DEL,SDFLAG(R0);ARE WE DELETING THE FILE?
23                 BEQ     8$            ;BR IF NOT
24         ; WE ARE FINISHED DELETING A FILE.
25         ; OUTPUT A CR-LF TO MAKE SURE THE PRINTER BUFFER GETS EMPTIED.
26         CALL    SWCRLF                ;WRITE CR-LF TO SPOOLED DEVICE
27         ; Call routine to see if we need to signal EOF to device handler
28         8$:  MOV     #DF$CLS,R1        ;Get .CLOSE special function code
29                 CALL    SDFUNC        ;See if we need to signal EOF to device
30         ; RELEASE ANY DISK BLOCKS THAT WE ARE HOLDING FOR BACKUP
31                 MOV     R2,-(SP)
32                 MOV     R0,R2          ;POINT TO CELLS HOLDING ADDRESSES
33                 ADD     #SDBU,R2
34         6$:  MOV     (R2)+,R1          ;GET SAVED DISK ADDRESS
35                 BEQ     7$            ;BR IF REACHED END OF LIST
36                 CLR     -2(R2)        ;CLEAR THE TABLE
37                 CALL    RLSBLK        ;RELEASE THE DISK BLOCK
38                 BR      6$
39         7$:  MOV     (SP)+,R2
40         ; RELEASE DEVICE AND FILE.
41         1$:  MOV     SDSFCB(R0),R1    ;POINT TO ACTIVE SFCB
42                 BIC     #SD$CLR,SDFLAG(R0);CLEAR SOME DEVICE CONTROL FLAGS
43                 CLR     SDSKIP(R0)    ;FINISHED SKIPPING
44                 CLR     SDBLK(R0)
45                 CLR     SDSFCB(R0)
46                 INC     NESB          ;ONE MORE EXTRA BUFFER AVAILABLE
47         ; Free the SFCB
48         CALL    FRESFB                ;Free the SFCB
49         ; TRY TO START UP NEXT FILE ON THIS DEVICE
50         4$:  CALL    SPOLGO          ;TRY TO START SPOOLER
51         RETURN

```

1				. SBTTL SWCRLF -- Write cr-lf to spooled device
2				-----
3				; SWCRLF is called to write carriage-return, line-feed to a spooled device.
4				;
5				; Inputs:
6				; R0 = Address of SDCB for the device.
7				;
8				; Outputs:
9				; R0 is preserved.
10				;
11	003462	010046		SWCRLF: MOV R0,-(SP)
12	003464	010146		MOV R1,-(SP)
13	003466	010246		MOV R2,-(SP)
14	003470	010002		MOV R0,R2 ;GET ADDRESS OF SDCB
15				;
16				; Get system I/O queue element
17				;
18	003472	010001		MOV R0,R1 ;GET ADDRESS OF SDCB
19	003474	062701	000000G	ADD #SDCHAN,R1 ;POINT TO CHANNEL BLOCK FOR SPOOLED DEV IN SDCB
20	003500	004767	000000G	CALL GETSYQ ;GET SYSTEM I/O QUEUE
21				;
22				; Set up queue element for the write
23				;
24	003504	062761	000001 000000G	ADD #1,Q.BLKN(R1) ;SAY BLOCK # = 1
25	003512	012700	000010'	MOV #CRTXT,R0 ;GET ADDRESS OF BUFFER TO BE WRITTEN
26	003516	004767	000000G	CALL SYBFAD ;STORE BUFFER ADDRESS IN I/O QUEUE ELEMENT
27	003522	012761	177777 000000G	MOV #-1,Q.WCNT(R1) ;SET WORD COUNT (- ==> WRITE)
28	003530	005061	000000G	CLR Q.COMP(R1) ;NO COMPLETION ROUTINE
29				;
30				; Start the I/O
31				;
32	003534	004767	000000G	CALL SYQID ;QUEUE THE I/O REQUEST
33				;
34				; Finished
35				;
36	003540	012602		MOV (SP)+,R2
37	003542	012601		MOV (SP)+,R1
38	003544	012600		MOV (SP)+,R0
39	003546	000207		RETURN

```

1          .SBTTL  SDFUNC -- Send special function code to spooled device
2          ;-----
3          ; SDFUNC is called to simulate a .SPFUN special function operation to
4          ; the spooled device. This is used to simulate a .ENTER or .CLOSE
5          ; operation to devices that accept these functions.
6          ;
7          ; Inputs:
8          ;   R1 = .SPFUN function code to be sent to handler.
9          ;   R0 = Address of SDCB
10         ;
11 003550 010046 SDFUNC: MOV     R0,-(SP)
12 003552 010246      MOV     R2,-(SP)
13 003554 010102      MOV     R1,R2          ; Carry function code in R2
14         ;
15         ; See if spooled device is directory structured and can accept .SPFUN
16         ; requests.
17         ;
18 003556 116001 000000G      MOVB   SDDVU(R0),R1      ; Get index number for spooled device
19 003562 032761 000000G 000000G      BIT    #DS$NRD,DVSTAT(R1) ; Special directory structured device?
20 003570 001004      BNE     1$          ; Br if yes
21 003572 032761 000000G 000000G      BIT    #DX$NRD,DVFLAG(R1) ; Hidden special dir dev flag set?
22 003600 001411      BEQ     9$          ; Br if not
23         ;
24         ; Send special function to handler
25         ;
26 003602 010001 1$:      MOV     R0,R1          ; Get address of SDCB
27 003604 062701 000000G      ADD     #SDCHAN,R1      ; Point to channel block in SDCB
28 003610 004767 000000G      CALL    GETSYQ        ; Get an I/O queue element
29 003614 110261 000000G      MOVB   R2,Q.FUNC(R1)    ; Set .SPFUN function code
30 003620 004767 000000G      CALL    SYQID         ; Queue the operation for the handler
31         ;
32         ; Finished
33         ;
34 003624 012602 9$:      MOV     (SP)+,R2
35 003626 012600      MOV     (SP)+,R0
36 003630 000207      RETURN

```

```

1          .SBTTL  SDCLOS -- Close spooled channel
2          ;-----
3          ; SDCLOS IS CALLED TO SEE IF A CHANNEL WHICH IS BEING
4          ; CLOSED IS ASSOCIATED WITH A SPOOLED DEVICE.
5          ;
6          ; Inputs:
7          ;   CHNNUM = User channel number.
8          ;   CHNADR = Address of current channel block.
9          ;   CORUSR = Job index number
10         ;
11 003632 010146      SDCLOS: MOV      R1,-(SP)
12 003634 010246      MOV      R2,-(SP)
13 003636 016701 000000G      MOV      CHNADR,R1      ;GET ADDRESS OF CURRENT CHANNEL BLOCK
14 003642 032761 000000G 000000G      BIT      #CS$SPL,C.CSW(R1); IS CHANNEL OPENED TO SPOOLED DEVICE?
15 003650 001427      BEQ      9$      ;BR IF NOT
16 003652 004767 001044      CALL     FNDSO      ;IS THIS A SPOOLED CHANNEL?
17 003656 103424      BCS      9$      ;BRANCH IF NOT
18          ; THIS IS A SPOOLED CHANNEL -- CLOSE OUT THE FILE.
19 003660 010046      MOV      R0,-(SP)
20 003662 016100 000000G      MOV      SFSDCB(R1),R0 ;GET ADDRESS OF SDCB
21          ; WRITE BLOCK WITH FLINK = 0 TO SIGNAL END OF FILE
22 003666 004767 175340      CALL     SGETBF      ;MAKE SURE WE'VE GOT BUFFER
23 003672 005067 000000G      CLR      SPUBUF      ;SET FILE FLINK = 0
24 003676 004767 175364      CALL     SFLUSH      ;WRITE OUT BUFFER
25 003702 005067 000000G      CLR      SX$FCB      ;SAY BUFFER IS FREE
26 003706 112761 177777 000000G      MOVB     #-1,SFCHAN(R1) ;DISASSOCIATE FILE AND CHANNEL
27          ; TRY TO START SPOOLER
28 003714 142761 000000G 000000G      BICB     #SF$HLD,SFFLAG(R1) ;SAY FILE IS FREE TO START
29 003722 004767 175616      CALL     SPOLGO      ;TRY TO START SPOOLER
30 003726 012600      MOV      (SP)+,R0
31 003730 012602      9$: MOV      (SP)+,R2
32 003732 012601      MOV      (SP)+,R1
33 003734 000207      RETURN

```

```

1          .SBTTL  SPLDEL -- Delete a spool file
2          ;-----
3          ; SPLDEL is called to delete a spool file.
4          ; There are three cases to contend with:
5          ; 1. Spool file is currently being processed by spooler.
6          ;    -- Set SD$DEL flag to cause spooler to delete rest of file.
7          ; 2. Spool file is still open.
8          ;    -- Set SF$DEL flag to cause spooler to delete file when it is closed.
9          ; 3. Spool file is closed but spooler hasn't started working on it yet.
10         ;    -- Delete the spool file immediately.
11         ;
12         ; Inputs:
13         ;   R3 = ID number of spool file to delete.
14         ;
15         ; Outputs:
16         ;   C-flag cleared ==> Spool file found and taken care of.
17         ;   C-flag set    ==> Could not find file with specified ID.
18         ;
19 003736 010146 SPLDEL: MOV     R1, -(SP)
20 003740 010246         MOV     R2, -(SP)
21 003742 010346         MOV     R3, -(SP)
22         ;
23         ; Try to find the spool file with the specified ID
24         ;
25 003744 012700 000000G         MOV     #SDCB, R0          ;Point to first SDCB
26 003750 020027 000000G 5$:    CMP     R0, #SDCBND        ;Checked all?
27 003754 103014             BHIS    6$                    ;Br if yes -- could not find file
28 003756 016002 000000G         MOV     SDFHD(R0), R2      ;Get address of 1st SFCB for this device
29 003762 001406             BEQ     7$                    ;Br if none
30 003764 020362 000000G 8$:    CMP     R3, SFID(R2)        ;Is this the SFCB for the correct file?
31 003770 001410             BEQ     10$                   ;Br if yes
32 003772 016202 000000G         MOV     SFQLNK(R2), R2     ;Get address of next SFCB for this device
33 003776 001372             BNE     8$                    ;Loop if more to check
34 004000 062700 000000G 7$:    ADD     #SDCBSZ, R0         ;Point to next SDCB
35 004004 000761             BR      5$                    ;Go check it
36         ;
37         ; Error: We cannot find a spool file with the specified ID
38         ;
39 004006 000261 6$:          SEC                      ;Signal error on return
40 004010 000457             BR      9$
41         ;
42         ; Found spool file with the specified ID
43         ;   R0 = SDCB address
44         ;   R2 = SFCB address
45         ; Set flag in SFCB saying file is to be deleted
46         ;
47 004012 152762 000000G 000000G 10$:  BISB    #SF$DEL, SFFLAG(R2) ;Set delete-file flag
48         ;
49         ; If file is currently being processed by the spooler, set the
50         ; delete-file flag for the spooler.
51         ;
52 004020 020260 000000G         CMP     R2, SDSFCB(R0)     ;Is this file the active one for this device?
53 004024 001004             BNE     1$                    ;Br if not
54 004026 052760 000000G 000000G         BIS     #SD$DEL, SDFLAG(R0) ;Tell spooler to delete the file
55 004034 000444             BR      11$
56         ;
57         ; File is not currently being processed by the spooler.

```

```

58      ; See if file has been closed yet.
59
60 004036 126227 000000G 177777 1$:      CMPB      SFCHAN(R2), #-1 ; Is file still open?
61 004044 001040      BNE      11$      ; Br if yes -- spooler will delete it later
62
63      ; File is closed and is not yet being processed by the spooler.
64      ; We will delete the file and free its blocks.
65
66 004046 152762 000000G 000000G      BISB      #SF$HLD, SFFLAG(R2) ; Set flag so spooler won't start on it
67
68      ; Get number of 1st block in this spool file
69
70 004054 016201 000000G      MOV      SFSTRT(R2), R1 ; Get # of 1st block in spool file
71
72      ; Read in 1st word of the block to get the flink to the next block
73
74 004060 010046 2$:      MOV      RO, -(SP) ; Save SDCB address
75 004062      READW      #SPLARG, #USPLCH, #SPDLBF, #1., R1 ; Read 1st word from block
76 004120 012600      MOV      (SP)+, RO ; Recover SDCB address
77
78      ; Free the block
79
80 004122 004767 000372      CALL      RLSBLK ; Release this block
81
82      ; Get number of next block in file
83
84 004126 016701 000000G      MOV      SPDLBF, R1 ; Get number of next block
85 004132 010162 000000G      MOV      R1, SFSTRT(R2) ; Set as starting block in case we are aborted
86 004136 001350      BNE      2$      ; Loop if more blocks to delete
87
88      ; We have finished freeing all of the blocks in the file.
89      ; Release the SFCB.
90
91 004140 010201      MOV      R2, R1 ; Get address of SFCB to R1 for FRESFB
92 004142 004767 000012      CALL      FRESFB ; Free the SFCB
93
94      ; We successfully deleted the file
95
96 004146 000241 11$:      CLC ; Signal success on return
97
98      ; Finished
99
100 004150 012603 9$:      MOV      (SP)+, R3
101 004152 012602      MOV      (SP)+, R2
102 004154 012601      MOV      (SP)+, R1
103 004156 000207      RETURN

```

```

1          .SBTTL  FRESFB -- Release a SFCB
2          ;-----
3          ; FRESFB is called to release a SFCB after we have finished processing
4          ; the spool file.
5          ;
6          ; Inputs:
7          ; R1 = Address of the SFCB
8          ;
9 004160 010046 FRESFB: MOV      R0,-(SP)
10 004162 010246      MOV      R2,-(SP)
11          ;
12          ; Get address of associated SDCB
13          ;
14 004164 016100 000000G      MOV      SFSDCB(R1),R0
15          ;
16          ; Clean out some cells in the SFCB
17          ;
18 004170 105061 000000G      CLRB      SFUSER(R1)      ;No associated user
19 004174 005061 000000G      CLR       SFSDCB(R1)      ;No associated SDCB
20          ;
21          ;
22          ; Remove this SFCB from the list associated with the SDCB
23          ; and put the SFCB on the free list.
24          ;
25 004200 010002      MOV      R0,R2      ;Point to SDCB
26 004202 062702 000000G  ADD      #SDFHD,R2      ;Point to list head for this SDCB
27 004206      DISABL      ;** Disable interrupts **
28 004214 020112 1$:      CMP      R1,(R2)      ;Is next entry the one we want?
29 004216 001404      BEQ      2$      ;Br if yes
30 004220 011202      MOV      (R2),R2      ;Chain to next one
31 004222 062702 000000G  ADD      #SFQLNK,R2      ;Point to flink field
32 004226 000772      BR       1$      ;Go check next one
33 004230 016112 000000G 2$:      MOV      SFQLNK(R1),(R2) ;Remove SFCB from chain
34 004234 016761 000000G 000000G  MOV      SFCBFH,SFQLNK(R1);Add SFCB to free list
35 004242 010167 000000G      MOV      R1,SFCBFH
36 004246 020167 000000G      CMP      R1,SDSFCB      ;Is our SFCB the current SFCB?
37 004252 001002      BNE      3$      ;Br if not
38 004254 005067 000000G      CLR      SDSFCB      ;Say no current SFCB
39 004260      ENABL      ;** Enable interrupts **
40          ;
41          ; Restart any users waiting for a free SFCB
42          ;
43 004266 012700 000000G      MOV      #S$SPCB,R0      ;Get wait state
44 004272 004767 000000G      CALL     UREGO      ;Restart waiting users
45          ;
46          ; Finished
47          ;
48 004276 012602      MOV      (SP)+,R2
49 004300 012600      MOV      (SP)+,R0
50 004302 000207      RETURN

```

```

1          .SBTTL  GTSBLK -- Get free spool file disk block
2          ;-----
3          ; GTSBLK IS CALLED TO GET A FREE SPOOL FILE DISK BLOCK.
4          ; IF NONE ARE AVAILABLE THE USER IS SUPENDED UNTIL
5          ; ONE IS RELEASED. WHEN CALLED, R1 MUST POINT TO
6          ; THE SFCB. ON RETURN THE FREE DISK BLOCK NUMBER
7          ; IS IN R0. ALL OTHER REGISTERS ARE PRESERVED.
8          ;
9 004304    010246    GTSBLK: MOV      R2,-(SP)
10 004306    010346    MOV      R3,-(SP)
11 004310
12          5$:      DISABL                      ;** DISABLE **
13          ; SEE IF THERE ARE ANY FREE PUBLIC SPOOL BLOCKS.
14          TST      NFRESB                      ;ANY FREE PUBLIC BLOCKS?
15          BNE      6$                          ;IF YES THEN GO USE ONE
16          ; THERE ARE NO FREE PUBLIC SPOOL BLOCKS.
17          ; EACH SPOOL DEVICE HAS A SMALL NUMBER OF PRIVATE BLOCKS.
18          ; SEE IF WE CAN FIND A FREE PRIVATE BLOCK TO USE.
19          MOV      SFSDCB(R1),R0              ;POINT TO SDCB
20          MOV      SDFRBL(R0),R2              ;GET # FREE PRIVATE BLOCKS
21          BEQ      1$                          ;GIVE UP IF NONE AVAIL
22          ; SAVE THE LAST FEW PRIVATE BLOCKS FOR ACTIVE FILES.
23          CMP      R2,#<SNBUX+2>              ;ENOUGH FOR INACTIVE FILES?
24          BHI      9$                          ;BR IF YES
25          BITB     #SF$BSY,SFFLAG(R1); IS THIS FILE ACTIVE?
26          BEQ      1$                          ;CAN'T GET BLOCK IF INACTIVE
27          9$:      DEC      SDFRBL(R0)          ;USE A PRIVATE BLOCK
28          BR       8$
29          ; THERE IS A FREE BLOCK -- FIND IT.
30          6$:      DEC      NFRESB              ;SAY ONE LESS BLOCK THERE
31          8$:      MOV      GRNTBL,R0           ;POINT TO BIT MAP TABLE
32          7$:      TST      (R0)+              ;ANY FREE BLOCKS IN THIS WORD?
33          BEQ      7$                          ;BRANCH IF NOT
34          MOV      -(R0),R2                    ;GET BITS FROM WORD
35          MOV      R0,-(SP)                    ;SAVE WORD LOCATION
36          SUB      GRNTBL,R0                    ;GET BYTE INDEX
37          ASH      #3,R0                        ;CONVERT BYTE TO BIT INDEX
38          CLR      R3                          ;COUNT BITS
39          3$:      ROL      R2                  ;SEARCH FOR FREE GRAN BIT
40          BCS      2$                          ;BRANCH IF FREE BIT FOUND
41          INC      R3                          ;KEEP TRACK OF ADDRESS
42          BR       3$
43          2$:      ADD      R3,R0              ;GET FINAL BLOCK ADDRESS
44          CLC                                          ;MARK BLOCK AS USED NOW
45          4$:      ROR      R2
46          DEC      R3
47          BGE      4$
48          MOV      R2,@(SP)+                    ;PUT BACK BIT MAP WORD
49          ENABL                      ;** ENABLE **
50          INC      R0                          ;1ST BLOCK IN FILE IS # 1
51          MOV      (SP)+,R3
52          MOV      (SP)+,R2
53          RETURN
54          ; THERE ARE NO FREE BLOCKS -- SUSPEND USER UNTIL ONE IS RELEASED.
55          ; DON'T HOLD FILE FOR CHANNEL CLOSE IF FILE IS FULL.
56          1$:      BITB     #SF$HLD,SFFLAG(R1) ; IS FILE BEING HELD?
57          BEQ      10$                          ;BR IF NOT
58          BICB     #SF$HLD,SFFLAG(R1); RELEASE THE FILE

```

58 004474	004767	175044	CALL	SPOLGO	; TRY TO START PRINTING FILE
59 004500	000703		BR	5\$	; NOW GO TRY TO GET SPACE AGAIN
60					
61 004502	012700	000000G			; SUSPEND USER UNTIL A BLOCK IS FREED
62 004506	004767	000000G	10\$: MOV	#S\$SPDB,RO	; QUEUE FOR SPOOL BLOCKS
63 004512	004767	000000G	CALL	QNSPND	; SUSPEND USER ** ENABLE **
64 004516	000674		CALL	CHKABT	; SEE IF WE'VE BEEN ABORTED
			BR	5\$	

```

1          .SBTTL  RLSBLK -- Free spool file disk block
2          ;-----
3          ; RLSBLK IS CALLED TO RELEASE A SPOOL FILE DISK BLOCK.
4          ; WHEN CALLED, R0 MUST POINT TO THE SDCB.
5          ; WHEN CALLED R1 MUST CONTAIN THE SPOOL FILE BLOCK NUMBER.
6          ; THE FIRST USER WAITING FOR A BLOCK IS RESTARTED BY RLSBLK.
7          ; ALL REGISTERS ARE PRESERVED EXCEPT R1.
8          ;
9 004520 010246 RLSBLK: MOV      R2, -(SP)
10 004522 010046      MOV      R0, -(SP)
11 004524 005301      DEC      R1          ; 1ST BLOCK IS # 1
12 004526 010100      MOV      R1, R0      ; SAVE DISK BLOCK #
13 004530 042700 177760 BIC      #177760, R0 ; GET # OF BIT IN WORD
14 004534 005002      CLR      R2
15 004536 000261      SEC
16 004540 006002 1$:  ROR      R2          ; SET BUSY BIT
17 004542 005300      DEC      R0          ; FORM MASK FOR SINGLE BIT
18 004544 002375      BGE      1$
19 004546 072127 177774 ASH      #-4, R1   ; GET WORD INDEX
20 004552 006301      ASL      R1
21 004554 066701 173222 ADD      GRNTBL, R1   ; POINT INTO BIT TABLE
22 004560 020167 173220 CMP      R1, GRNEND  ; MAKE SURE STORE IS OK
23 004564 103401      BLO      2$          ; BRANCH IF OK
24 004566 000000      HALT
25 004570 050211 2$:  BIS      R2, (R1)     ; SAFETY STOP
26          ; INCREMENT NUMBER OF FREE PUBLIC OR PRIVATE BLOCKS
27 004572 011600      MOV      (SP), R0    ; MARK BLOCK AS FREE
28 004574 026027 000000G 000000G      CMP      SDFRBL(R0), #PVSPBL ; POINT TO SDCB
29 004602 103003      BHIS     3$          ; ADD FREE BLOCK TO PUBLIC OR PRIVATE?
30 004604 005260 000000G      INC      SDFRBL(R0) ; BR IF PUBLIC
31 004610 000402      BR       4$          ; ADD TO PRIVATE
32 004612 005267 000000G 3$:  INC      NFRESB ; SAY ANOTHER FREE BLOCK
33          ; RESTART ANY USER WAITING FOR DISK BLOCK
34 004616 012700 000000G 4$:  MOV      #S$SPDB, R0 ; POINT TO WAIT QUEUE HEAD
35 004622 004767 000000G      CALL     UREGO  ; REACTIVATE 1ST USER
36 004626 012600      MOV      (SP)+, R0
37 004630 012602      MOV      (SP)+, R2
38 004632 000207      RETURN
39

```

```

1          .SBTTL  RLSBUF -- Release spool buffer
2          ;-----
3          ; RLSBUF IS CALLED TO RELEASE A SPOOL CORE BUFFER.
4          ; WHEN CALLED, R1 MUST POINT TO THE BUFFER.
5          ; SPOLRD IS CALLED TO RESTART ANY SPOOLERS WHO
6          ; ARE WAITING FOR A BUFFER.
7          ; ALL REGISTERS ARE PRESERVED EXCEPT R1.
8          ;
9 004634 010046 RLSBUF: MOV      RO, -(SP)
10 004636      DISABL      ;** DISABLE **
11          ; RELEASE THE BUFFER
12 004644 016711 000000G      MOV      SPLBHD, (R1)      ; LINK INTO FREE CHAIN
13 004650 010167 000000G      MOV      R1, SPLBHD
14          ; RESTART ANY WAITING SPOOL BUFFER REQUESTS.
15 004654 016700 173126      MOV      SPBWH, RO      ; HEAD OF WAIT LIST
16 004660 001413      BEQ      1$      ; BR IF NOONE WAITING
17 004662 016067 000000G 173116      MOV      SDWLST(RO), SPBWH      ; REMOVE 1ST FROM LIST
18 004670 042760 000000G 000000G      BIC      #SD$BWT, SDFLAG(RO) ; SAY NO LONGER WAITING FOR BUFFER
19 004676      ENABL      ;** ENABLE **
20 004704 004767 175440      CALL      SPOLRD      ; SAY IT CAN DO A READ
21 004710      1$: ENABL      ;** ENABLE **
22 004716 012600      MOV      (SP)+, RO
23 004720 000207      RETURN

```

```

1          .SBTTL  FNDSO -- Locate SFCB for spooled channel
2          ;-----
3          ; FNDSO is called to locate the SFCB associated with the current I/O channel.
4          ;
5          ; Inputs:
6          ;   CHNNUM = User channel number.
7          ;   CORUSR = Job index number
8          ;
9          ; Outputs:
10         ;   C-flag cleared if SFCB found, set otherwise.
11         ;   R1 = Address of SFCB.
12         ;
13 004722 016701 000000G  FNDSO:  MOV     SFCB,R1          ;POINT TO 1ST SFCB
14 004726 020167 000000G  4$:    CMP     R1,SFCBND        ;CHECKED ALL SFCB'S?
15 004732 103013          ;      BHIS     1$              ;BR IF YES
16 004734 126761 000000G 000000G  ;      CMPB    CORUSR,SFUSER(R1); IS THIS SFCB IN USE BY THIS USER?
17 004742 001004          ;      BNE     2$              ;BR IF NOT
18 004744 126761 000000G 000000G  ;      CMPB    CHNNUM,SFCHAN(R1); IS SFCB ASSIGNED TO THIS CHANNEL?
19 004752 001405          ;      BEQ     3$              ;BR IF YES
20 004754 062701 000000G  2$:    ADD     #SFCBSZ,R1        ;POINT TO NEXT SFCB
21 004760 000762          ;      BR      4$              ;GO CHECK IT
22          ; Can't find SFCB.
23 004762 000261 1$:    SEC                      ;SIGNAL ERROR ON RETURN
24 004764 000401          ;      BR      5$
25          ; Found the SFCB.
26 004766 000241 3$:    CLC                      ;SIGNAL SUCCESS ON RETURN
27 004770 000207 5$:    RETURN

```

```

1          .SBTTL  CVTDVU -- Convert device name to dev index and unit #
2          ;-----
3          ; CVTDVU is called to convert a RAD50 device name into the corresponding
4          ; device index number and unit number.
5          ;
6          ; Inputs:
7          ;   R1 = RAD50 device name.
8          ;
9          ; Outputs:
10         ;   C-flag cleared ==> Conversion successful.
11         ;   C-flag set      ==> Unable to find device name in tables.
12         ;   R1 = Device index number (low byte), device unit number (high byte).
13         ;
14 004772 010246 CVTDVU: MOV      R2,-(SP)
15 004774 010346      MOV      R3,-(SP)
16
17         ; Split the unit number off of the full device name
18         ;
19 004776 010103      MOV      R1,R3          ;Get full device name
20 005000 005002      CLR      R2            ;Set up for divide
21 005002 071227 000050 DIV      #50,R2      ;Split name and unit (R2=name, R3=unit)
22 005006 005703      TST      R3            ;Was a unit number specified?
23 005010 001402      BEQ      1$            ;Br if not
24 005012 162703 000036 SUB      #36,R3      ;Convert unit number to binary value
25 005016 010301 1$:  MOV      R3,R1          ;Get unit number
26 005020 000301      SWAB     R1            ;Position to high-order byte
27
28         ; Look up the device name to get the device index
29         ;
30 005022 070227 000050      MUL      #50,R2      ;Now get the device name without unit number
31 005026 016702 000000G MOV      NUMDEV,R2      ;Get index number of last device
32 005032 020362 000000G 2$:  CMP      R3,PNAME(R2)    ;Search for device in name table
33 005036 001407      BEQ      3$            ;Br if found it
34 005040 162702 000002      SUB      #2,R2      ;Try next device
35 005044 002372      BGE      2$            ;Loop if more to check
36
37         ; Error, cannot find device name in tables
38         ;
39 005046 012701 177777      MOV      #-1,R1      ;Set device # = unit # = -1
40 005052 000261      SEC              ;Signal error on return
41 005054 000402      BR       9$
42
43         ; Found the device in the tables
44         ;
45 005056 050201 3$:  BIS      R2,R1          ;Combine device # and unit #
46 005060 000241      CLC              ;Signal success on return
47
48         ; Finished
49         ;
50 005062 012603 9$:  MOV      (SP)+,R3
51 005064 012602      MOV      (SP)+,R2
52 005066 000207      RETURN

```

```

1          .SBTTL  EMSPHL -- EMT to set spool HOLD mode
2          ;-----
3          ; Kmon EMT to set spool hold mode.
4          ;
5          ; Inputs:
6          ;   EMT arg block+4 = Address of SDCB for spooled device.
7          ;
8 005070    EMSPHL:
9          ;
10         ; Set hold flag in SDCB
11         ;
12 005070    016702    000004G          MOV      EMTBLK+4,R2      ;Point to SDCB
13 005074    052762    000000G 000000G    BIS      #SD$HLD,SDFLAG(R2);Set hold flag
14         ;
15         ; Set SF$HLD flag in each spool file
16         ;
17 005102    016203    000000G          MOV      SDFHD(R2),R3      ;Point to 1st SFCB for the device
18 005106    001406          BEQ          9$                      ;Br if there are none
19 005110    152763    000000G 000000G 1$:  BISB     #SF$HLD,SFFLAG(R3);Set hold flag for file
20 005116    016303    000000G          MOV      SFQLNK(R3),R3      ;Get address of next SFCB
21 005122    001372          BNE          1$                      ;Br if there is another file
22         ;
23         ; Finished
24         ;
25 005124    000167    000000G    9$:      JMP      EMTXIT

```

```

1          .SBTTL  EMSPNH -- EMT to set spool NOHOLD mode
2          ;-----
3          ; Kmon EMT to set spool nohold mode.
4          ;
5          ; Inputs:
6          ;   EMT arg block+4 = Address of SDCB for spooled device.
7          ;
8 005130    EMSPNH:
9          ;
10         ;   Reset hold flag in SDCB
11         ;
12 005130    016702    0000040      MOV      EMTBLK+4,R2      ;Point to SDCB
13 005134    042762    0000000 0000000      BIC      #SD$HLD,SDFLAG(R2);Reset hold flag
14         ;
15         ;   Clear SF$HLD flag in each spool file
16         ;
17 005142    016203    0000000      MOV      SDFHD(R2),R3      ;Point to 1st SFCB for the device
18 005146    001411      BEQ      9$      ;Br if there are none
19 005150    142763    0000000 0000000 1$:    BICB      #SF$HLD,SFFLAG(R3);Clear hold flag for file
20 005156    016303    0000000      MOV      SFQLNK(R3),R3      ;Get address of next SFCB
21 005162    001372      BNE      1$      ;Br if there is another file
22         ;
23         ;   Try to start spooler
24         ;
25 005164    010200      MOV      R2,R0      ;Get address of SDCB
26 005166    004767    174352      CALL      SPOLGO
27         ;
28         ;   Finished
29         ;
30 005172    000167    0000000      9$:    JMP      EMTXIT
  
```

```

1          .SBTTL  SFLGPG  -- COPY FLAG PAGE TO SPOOL BUFFER
2          ;-----
3          ; SFLGPG copies the contents of the stolen CSI buffer into the
4          ; spool buffer.
5          ;
6 005176 010246      SFLGPG: MOV      R2,-(SP)          ;SAVE R2
7
8 005200 012704 0000000      MOV      #CSIBUF,R4          ;GET BEGINNING OF STOLEN BUFFER
9 005204 016702 0000000      MOV      SXBPNT,R2          ;GET POINTER INTO SPOOL BUFFER
10 005210 020227 0000000      1$:  CMP      R2,#SPUBND      ;BUFFER FULL?
11 005214 103406          BLO      2$          ;BR IF NOT
12 005216 010267 0000000      MOV      R2,SXBPNT          ;SAVE POINTER INTO SPOOL BUFFER
13 005222 004767 173732      CALL     SBUFOT          ;WRITE TO SPOOL FILE
14 005226 016702 0000000      MOV      SXBPNT,R2          ;RESTORE POINTER INTO SPOOL BUFFER
15 005232 112422          2$:  MOVB     (R4)+,(R2)+      ;MOVE INTO SPOOL BUFFER
16 005234 121427 000200      CMPB     (R4),#200          ;END?
17 005240 001363          BNE      1$          ;BR IF NOT
18
19 005242 010267 0000000      MOV      R2,SXBPNT          ;SAVE POINTER INTO SPOOL BUFFER
20 005246 012602          MOV      (SP)+,R2          ;RESTORE R2
21 005250 000207          RETURN
22
23          ; Top of TSSPOL module
24          ;
25 005252          SPLTOP:
26          .END

```

Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 9631 Words (38 Pages)
Size of core pool: 18176 Words (71 Pages)
Operating system: RT-11

Elapsed time: 00:00:42.53
,LP:TSSPOL=DK:TSSPOL/C/N:SYM

\$DILUP	1-58	9-90																	
. . . V1	7-28	7-28	7-30	18-75	18-75														
. . . V2	7-28	7-28	7-28#	7-28#	18-75	18-75	18-75#	18-75#											
BELL	1-94#																		
C. CSW	1-63	3-106*	17-14																
CHKABT	1-66	3-50	20-63																
CHKSD	1-44	3-21#																	
CHNADR	1-63	3-105	17-13																
CHNNUM	1-76	3-84	23-18																
CORUSR	1-52	3-60	3-72	23-16															
CR	1-88	1-92#	4-62																
CRTXT	1-88#	15-25																	
CS\$SPL	1-62	3-106	17-14																
CSIBND	1-52																		
CSIBUF	1-52	27-8																	
CTRLTT	1-71	9-88	9-112																
CVTDVU	3-25	24-14#																	
DF\$CLS	1-50	14-28																	
DF\$ENT	1-50	9-128																	
DYDEV	1-55	3-29																	
DS\$NRD	1-50	16-19																	
DVFLAG	1-51	16-21																	
DVSTAT	1-50	16-19																	
DX\$NRD	1-51	16-21																	
EMSPHL	1-44	25-8#																	
EMSPNH	1-44	26-8#																	
EMTBK	1-51	4-27	4-28	4-42	25-12	26-12													
EMTXIT	1-50	25-25	26-30																
FILSPC	1-62	3-24	3-39	3-40															
FLGPAG	1-59	4-83																	
FMMSG	1-59	9-96																	
FNDSD	4-18	8-14	17-16	23-13#															
FRESFB	14-48	18-92	19-9#																
FREUSR	1-71	3-42																	
GETSYQ	1-77	10-70	13-37	15-20	16-28														
GETUCH	1-76	4-49	4-61	4-71	4-93														
GRNEND	1-86#	2-21*	21-22																
GRNTBL	1-85#	2-14*	20-30	20-35	21-21														
GTSBK	3-99	5-26	20-9#																
GTSYMB	1-54	9-92																	

PSW	1-63	3-54*	3-58*	3-88*	3-97*	9-7*	9-83*	9-126*	9-147*	10-13*	10-45*	10-89*
	11-10*	11-23*	13-9*	13-19*	13-58*	13-65*	19-27*	19-39*	20-11*	20-48*	22-10*	22-19*
	22-21*											
PVSPBL	1-53	21-28										
Q. BLKN	1-77	10-76*	13-42*	15-24*								
Q. CHAN	1-78	10-75*	13-51*									
Q. COMP	1-78	10-80*	13-50*	15-28*								
Q. FUNC	1-50	16-29*										
Q. WCNT	1-78	10-79*	13-49*	15-27*								
QMSG	1-54	9-113										
QNSPND	1-66	3-49	20-62									
RLSBLK	12-20	14-37	18-80	21-9*								
RLSBUF	14-13	22-9*										
RSQBR	1-95*	4-51										
S\$SPCB	1-64	3-48	19-43									
S\$SPDB	1-64	9-143	20-61	21-34								
SB\$TXT	1-54	9-95	9-101									
SBUFOT	4-91	5-16*	6-17	27-13								
SBUFWD	1-73	10-79										
SD\$BAK	1-74	10-18	10-49	10-60								
SD\$BWT	1-59	10-16	10-37	22-18								
SD\$CLR	1-74	14-42										
SD\$DEL	1-73	9-133	13-20	14-22	18-54							
SD\$FLG	1-57	4-81										
SD\$FLK	1-57	9-57										
SD\$HLD	1-60	3-81	25-13	26-13								
SD\$INR	1-72	10-44	12-35	13-14								
SD\$RSV	1-74	8-10										
SD\$SMS	1-56	9-78	9-116									
SD\$SNG	1-56	9-76										
SD\$WFM	1-57	9-28	9-115									
SDANAM	1-57	9-107										
SDBLK	1-72	9-135*	9-138*	13-42	13-43*	14-44*						
SDBU	1-75	10-54	12-19	12-23	12-28	14-33						
SDBUF1	1-70	10-24	12-38	13-10	13-17*	14-11	14-14*					
SDBUF2	1-70	10-14	10-43*	12-34	13-12	13-18*						
SDBULS	1-75	12-16										
SDCB	1-51	3-26	18-25									
SDCBND	1-51	3-27	18-26									
SDCBSZ	1-73	3-33	18-34									
SDCHAN	1-69	13-36	15-19	16-27								
SDCLOS	1-44	17-11*										
SDDVU	1-61	3-31	16-18									
SDFHD	1-72	3-89	9-16	9-35	9-63	18-28	19-26	25-17	26-17			
SDFLAG	1-69	3-81	4-81	9-28	9-57	9-76	9-78	9-115*	9-116*	9-133*	10-16	10-18
	10-37*	10-44*	10-49	10-60*	12-35*	13-14	13-20	14-22	14-42*	18-54*	22-18*	25-13*
	26-13*											
SDFLNK	1-70	9-134*	10-22	10-55*	10-76	12-31	12-34*					
SDFORM	1-57	9-45	9-47	9-49								
SDFRBL	1-53	20-19	20-26*	21-28	21-30*							
SDFUNC	9-129	14-29	16-11*									
SDMOVE	1-46	4-10*										
SDMV1	4-37	4-75	4-79	4-82	4-87*							
SDMVX	4-19	4-30	4-104*									
SDNAME	1-55	3-29										
SDSFCB	1-70	9-8	9-124*	10-10	12-32	14-41	14-45*	18-52	19-36	19-38*		

SDSKIP	1-75	13-22	13-24*	14-43*								
SDWLST	1-71	11-14	11-18*	11-22*	22-17							
SF\$1ST	1-55	4-36	4-48	9-40								
SF\$BN1	1-52	4-44	4-78	9-136								
SF\$BSY	1-69	7-35	9-125	20-24								
SF\$DEL	1-55	9-43	9-131	18-47								
SF\$HLD	1-60	3-83	8-20	8-21	8-24	9-36	9-66	17-28	18-66	20-55	20-57	25-19
	26-19											
SFCB	1-65	2-40	23-13									
SFCBFH	1-74	2-41*	3-55	3-57*	19-34	19-35*						
SFCBND	1-65	23-14										
SFCBSZ	1-55	2-44	23-20									
SFCHAN	1-67	3-84*	17-26*	18-60	23-18							
SFFILE	1-67	3-69*	3-70*	3-73*	3-74*							
SFFLAG	1-67	3-61*	3-83*	4-36	4-44*	4-48*	4-78	7-35	8-19	8-21*	8-24*	9-36
	9-40	9-43	9-66	9-125*	9-131	9-136	17-28*	18-47*	18-66*	20-24	20-55	20-57*
	25-19*	26-19*										
SFFLNK	1-68	3-101*	7-28	7-31*								
SFFORM	1-56	3-77*	3-78*	3-79*	4-58	9-21	9-45	9-47	9-49	9-102		
SFID	1-58	3-67*	18-30									
SFLGPG	1-45	27-6#										
SFLUSH	5-29	7-11#	17-24									
SFNMBL	1-68	3-86*	7-33*	9-19	9-38	9-64	10-20	10-56*	12-33*			
SFQLNK	1-69	2-49*	3-57	3-93	3-96*	9-23	9-51	9-68	18-32	19-31	19-33	19-34*
	25-20	26-20										
SFRCMP	10-80	12-15#										
SFSDCB	1-68	3-85*	4-80	7-34	17-20	19-14	19-19*	20-18				
SFSTRT	1-68	3-100*	9-134	18-70	18-85*							
SFUSER	1-67	3-60*	19-18*	23-16								
SGETBF	4-23	6-11#	17-22									
SGFSF	9-50	9-76#										
SGNMT	9-67	9-83#										
SGOTFL	9-22	9-42	9-44	9-77	9-79	9-124#						
SGX1	9-18	9-30	9-58	9-70	9-120	9-146#						
SGX2	9-10	9-147#										
SNBUX	1-60	2-56	20-22									
SPBWHD	1-87#	10-38	22-15	22-17*								
SPDLBF	1-53	18-75	18-84									
SPLARG	1-62	7-28	18-75									
SPLBHD	1-65	2-28*	10-33	10-42*	22-12	22-13*						
SPLCHN	1-62	10-69										
SPLDEL	1-46	18-19#										
SPLERR	1-59	7-30										
SPLINI	1-45	2-7#										
SPLNB	1-52	2-26	10-29									
SPLND	1-52	2-56	10-29									
SPLSHF	1-45	8-9#										
SPLTOP	1-46	2-13	27-25#									
SPOLGO	1-45	7-39	9-7#	14-50	17-29	20-58	26-26					
SPOLID	1-58	3-63	3-65*	3-66*	3-67							
SPOLRD	7-37	9-140	10-8#	13-59	22-20							
SPOLWR	12-40	13-7#	14-19									
SPOOLQ	10-39	11-9#										
SPUBND	1-76	4-88	27-10									
SPUBUF	1-62	5-22	5-27*	6-20	7-15	17-23*						
SREX	10-15	10-17	10-21	10-23	10-30	10-89#						

SRX	10-11	10-40	10-90#									
SWCMP	13-50	13-64	14-10#									
SWCRLF	14-26	15-11#										
SXBPNT	1-61	4-87	4-90*	4-92	4-100*	5-21	6-20*	7-19	7-42*	27-9	27-12*	27-14
	27-19*											
SXSFCB	1-61	5-18	6-11	6-13	6-19*	7-16	17-25*					
SYBFAD	1-65	10-78	13-46	15-26								
SYQIO	1-78	10-85	13-55	15-32	16-30							
TSSPOL	1-13#	1-44										
UFORM	1-53	3-77	3-78	3-79								
URO	1-76	4-29*	8-10*	8-19*	8-20*							
UREGO	1-66	9-144	19-44	21-35								
USPLCH	1-64	7-28	7-28	18-75	18-75							

