

[illegible]

```
LL      AAAAAA  BBBB BBBB  UU      UU  FFFFFFFF  FFFFFFFF  EEEEEEEEE  RRRRRRRR
LL      AAAAAA  BBBB BBBB  UU      UU  FFFFFFFF  FFFFFFFF  EEEEEEEEE  RRRRRRRR
LL      AA      AA  BB      BB  UU      UU  FF      FF      EE      RR      RR
LL      AA      AA  BB      BB  UU      UU  FF      FF      EE      RR      RR
LL      AA      AA  BB      BB  UU      UU  FF      FF      EE      RR      RR
LL      AA      AA  BB      BB  UU      UU  FF      FF      EE      RR      RR
LL      AA      AA  BBBB BBBB  UU      UU  FFFFFFFF  FFFFFFFF  EEEEEEEEE  RRRRRRRR
LL      AA      AA  BBBB BBBB  UU      UU  FFFFFFFF  FFFFFFFF  EEEEEEEEE  RRRRRRRR
LL      AAAAAAAAAA  BB      BB  UU      UU  FF      FF      EE      RR      RR
LL      AAAAAAAAAA  BB      BB  UU      UU  FF      FF      EE      RR      RR
LL      AA      AA  BB      BB  UU      UU  FF      FF      EE      RR      RR
LL      AA      AA  BB      BB  UU      UU  FF      FF      EE      RR      RR
LL      AA      AA  BBBB BBBB  UU      UU  FF      FF      EE      RR      RR
LL      AA      AA  BBBB BBBB  UU      UU  FF      FF      EE      RR      RR
LLLLLLLLLL  AA      AA  BBBB BBBB  UUUUUUUUU  FF      FF      EEEEEEEEE  RR      RR
LLLLLLLLLL  AA      AA  BBBB BBBB  UUUUUUUUU  FF      FF      EEEEEEEEE  RR      RR
```

```
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
LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
```


(2)	47	DECLARATIONS
(3)	71	LPASIBFSTS - GET BUFFER STATUS
(4)	171	LPASIGTBUF - GET A BUFFER
(5)	240	LPASINXTBF - SET NEXT BUFFER
(5)	241	LPASRMVBUF - REMOVE BUFFER
(6)	333	LPASIWTBUF - WAIT FOR A BUFFER
(7)	436	LPASRLSBUF - RELEASE BUFFER
(8)	542	LPASSBFRST - BUFFER FULL/EMPTY AST HANDLER
(9)	619	LPASSOVRST - BFR OVER/UNDERRUN AST HANDLER
(10)	686	LPASSCMPLTAST - QIO COMPLETE AST HANDLER
(11)	763	CVTADINDX - CONVERT ADDRESS TO INDEX
(12)	806	GIVUSRBFR - GIVE USER BUFFER

```

0000 1      .TITLE LPA$BUFFER
0000 2      .IDENT 'V04-000'
0000 3
0000 4
0000 5      *****
0000 6      *
0000 7      *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8      *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9      *  ALL RIGHTS RESERVED.
0000 10     *
0000 11     *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12     *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13     *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14     *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15     *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16     *  TRANSFERRED.
0000 17     *
0000 18     *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19     *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20     *  CORPORATION.
0000 21     *
0000 22     *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23     *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24     *
0000 25     *
0000 26     *****
0000 27
0000 28
0000 29     ++
0000 30     FACILITY:      LPA-11 PROCEDURE LIBRARY
0000 31
0000 32     ABSTRACT:
0000 33     THIS MODULE CONTAINS THE ROUTINES WHICH MANIPULATE BUFFERS
0000 34     AS PART OF THE LPA-11 PROCEDURE LIBRARY. THIS INCLUDES THE
0000 35     AST ROUTINES.
0000 36
0000 37     ENVIRONMENT:  USER MODE, SHARED OR NON-SHARED LIBRARY
0000 38
0000 39     AUTHOR:  STEVE BECKHARDT,      CREATION DATE:  28-AUG-78
0000 40
0000 41     MODIFIED BY:
0000 42
0000 43     V03-001 SBL3001      Steven B. Lionel      30-March-1982
0000 44     Change module name to LPA$BUFFER.
0000 45     --

```


DECLARATIONS

```

0000 47          .SBTTL DECLARATIONS
0000 48 :
0000 49 : INCLUDE FILES:
0000 50 :
0000 51 :
0000 52 :
0000 53 : MACROS:
0000 54 :
0000 55 :
0000 56 :
0000 57 : EQUATED SYMBOLS:
0000 58 :
0000 59 :
0000 60 :
0000 61 : OWN STORAGE:
0000 62 :
0000 63 :
00000000 64          .PSECT _LPASCODE,PIC,SHR,EXE,NOWRT,LONG
0000 65
0000 66
0000 67 LPASBFSTART::
0000 68
0000 69

```

; START OF BUFFER ROUTINES (MAY BE
; USED TO LOCK THESE ROUTINES INTO
; THE PROCESS'S WORKING SET).

LPA\$IBFSTS - GET BUFFER STATUS

```
0000 71 .SBTTL LPA$IBFSTS - GET BUFFER STATUS
0000 72 :++
0000 73 : FUNCTIONAL DESCRIPTION
0000 74 :
0000 75 : THIS ROUTINE RETURNS THE STATUS OF EACH OF THE BUFFERS.
0000 76 : THE STATUS IS DEFINED AS WHICH QUEUE THE BUFFER IS ON.
0000 77 :
0000 78 : CALLING SEQUENCE:
0000 79 :
0000 80 : CALLS/G
0000 81 :
0000 82 : INPUT PARAMETERS:
0000 83 :
0000 84 : IBUF(AP) ADDRESS OF IBUF ARRAY
0000 85 : ISTAT(AP) ADDRESS OF LONGWORD ARRAY WITH AS MANY
0000 86 : ELEMENTS AS THERE ARE BUFFERS INVOLVED
0000 87 : IN THE SWEEP
0000 88 :
0000 89 : IMPLICIT INPUTS:
0000 90 :
0000 91 : VARIOUS FIELDS IN THE IBUF ARRAY
0000 92 :
0000 93 : OUTPUT PARAMETERS:
0000 94 :
0000 95 : ISTAT(AP) ADDRESS OF LONGWORD ARRAY WITH AS MANY
0000 96 : ELEMENTS AS THERE ARE BUFFERS INVOLVED
0000 97 : IN THE SWEEP. EACH ELEMENT IS FILLED WITH
0000 98 : THE STATUS OF THE CORRESPONDING BUFFER.
0000 99 : THE STATUS CODES ARE AS FOLLOWS:
0000 100 :
0000 101 : 2 BUFFER IS ON DEVICE QUEUE
0000 102 : 1 BUFFER IS ON USER QUEUE
0000 103 : 0 BUFFER IS NOT ON ANY QUEUE
0000 104 : -1 BUFFER IS ON INUSE QUEUE
0000 105 :
0000 106 : IMPLICIT OUTPUTS:
0000 107 :
0000 108 : NONE
0000 109 :
0000 110 : COMPLETION CODES:
0000 111 :
0000 112 : NONE
0000 113 :
0000 114 : SIDE EFFECTS:
0000 115 :
0000 116 : VARIOUS FIELDS IN THE IBUF ARRAY ARE MODIFIED
0000 117 :
0000 118 : --
0000 119 :
005C 0000 120 .ENTRY LPA$IBFSTS,^M<R2,R3,R4,R6>
0002 121
0002 122 MOVL 4(AP),R6 ; GET ADDRESS OF IBUF ARRAY
0006 123 MOVL 8(AP),R4 ; GET ADDRESS OF ISTAT ARRAY
000A 124
000A 125 EXTZV #0,#3,IBF$L_CMDTBL+CMT$B_VBFRMASK(R6),R0 ; GET HIGHEST BUF. #
0010 126
0010 127 ; INITIALIZE ALL ENTRIES IN ISTAT ARRAY TO 0
```

56 04 AC D0
54 08 AC D0
50 22 A6 03 00 EF

LPASIBFSTS - GET BUFFER STATUS

```

        6440  D4 0010 128 10$: CLRL (R4)[R0]
        FA 50  F4 0013 129  SOBGEQ R0,10$
                0016 130
                0016 131
                0016 132
                001F 133
                001F 134
        52 53 02 D0 001F 135
        58 A6 DE 0022 136
        1B 10 0026 137
                0028 138
                0028 139
        52 50 53 D7 0028 140
        A6 DE 002A 141
        13 10 002E 142
                0030 143
                0030 144
        52 53 01 CE 0030 145
        60 A6 DE 0033 146
        0A 10 0037 147
                0039 148
                0039 149
                0039 150
        04 0042 151
                0043 152
                0043 153
                0043 154
                0043 155
                0043 156
                0043 157
                0043 158
                0043 159
                0043 160 FOLLOWQ:
        51 52 D0 0043 161
                0046 162
        51 61 D0 0046 163 10$:
        52 51 D1 0049 164
                09 13 004C 165
        01EA 30 004E 166
        6440 53 D0 0051 167
        EF 11 0055 168
        05 0057 169 20$: RSB

; *** TURN OFF ASTS ***
$SETAST_S #0

; STORE STATUS FOR ALL BUFFERS ON THE DEVICE QUEUE
MOVL #2,R3 ; STATUS = 2
MOVAL IBF$L DEVQFL(R6),R2 ; ADDRESS OF DEVICE QUEUE
BSBB FOLLOWQ ; FOLLOW THE QUEUE

; STORE STATUS FOR ALL BUFFERS ON THE USER QUEUE
DECL R3 ; STATUS = 1
MOVAL IBF$L USRQFL(R6),R2 ; ADDRESS OF USER QUEUE
BSBB FOLLOWQ ; FOLLOW THE QUEUE

; STORE STATUS FOR ALL BUFFERS ON THE INUSE QUEUE
MNEGL #1,R3 ; STATUS = -1
MOVAL IBF$L INUQFL(R6),R2 ; ADDRESS OF INUSE QUEUE
BSBB FOLLOWQ ; FOLLOW THE QUEUE

; *** TURN ON ASTS ***
$SETAST_S #1
RET

; LOCAL SUBROUTINE TO FOLLOW QUEUE AND STORE STATUS IN ISTAT ARRAY
; INPUT: R2 CONTAINS ADDRESS OF QUEUE HEAD
; R3 CONTAINS STATUS TO STORE IN ISTAT ARRAY
; FOLLOWQ:
MOVL R2,R1 ; COPY ADDRESS OF QUEUE HEAD
; GET ADDRESS OF NEXT LINK
; BACK TO QUEUE HEAD YET?
; YES, DONE
; NO, CONVERT ADDRESS TO BUFFER INDEX
; STORE STATUS IN CORRESPONDING ENTRY
; OF ISTAT AND BRANCH BACK FOR NEXT ONE
```

LPASIGTBUF - GET A BUFFER

```

0058 171 .SBTTL LPASIGTBUF - GET A BUFFER
0058 172 :++
0058 173 : FUNCTIONAL DESCRIPTION:
0058 174 :
0058 175 : THIS ROUTINE IS CALLED BY THE USER'S PROGRAM TO GET A BUFFER
0058 176 : FROM THE HEAD OF THE USER QUEUE. UNLIKE LPASIWTFBUF, THIS ROUTINE
0058 177 : RETURNS IMMEDIATELY IF THE QUEUE IS EMPTY. THIS ROUTINE IS INTENDED
0058 178 : TO BE CALLED FROM THE USER'S COMPLETION ROUTINE (WHICH IS CALLED BY
0058 179 : THE AST HANDLERS). THIS ROUTINE SHOULD BE CALLED IF THE USER
0058 180 : IS SYNCHRONIZING WITH A COMPLETION ROUTINE, RATHER THAN WITH EVENT
0058 181 : FLAGS.
0058 182 :
0058 183 : CALLING SEQUENCE:
0058 184 :
0058 185 : CALLS/G
0058 186 : MAY ALSO BE CALLED AS A FUNCTION
0058 187 :
0058 188 : INPUT PARAMETERS:
0058 189 :
0058 190 : IBUF(AP) ADDRESS OF IBUF ARRAY
0058 191 : IBUFNO(AP) ADDRESS OF LONGWORD WHICH RECEIVES BUFFER
0058 192 : INDEX OR -1 WHICH INDICATES NO BUFFER IN QUEUE
0058 193 :
0058 194 : IMPLICIT INPUTS:
0058 195 :
0058 196 : VARIOUS FIELDS IN THE IBUF ARRAY
0058 197 :
0058 198 : OUTPUT PARAMETERS:
0058 199 :
0058 200 : IBUFNO(AP) ADDRESS OF LONGWORD WHICH RECEIVES BUFFER
0058 201 : INDEX OR -1 WHICH INDICATES NO BUFFER IN QUEUE
0058 202 :
0058 203 : IMPLICIT OUTPUTS:
0058 204 :
0058 205 : OFFSET IBF$Q_IOST IN THE IBUF ARRAY CONTAINS ADDITIONAL STATUS
0058 206 :
0058 207 : COMPLETION CODES:
0058 208 :
0058 209 : NONE
0058 210 :
0058 211 : SIDE EFFECTS:
0058 212 :
0058 213 : VARIOUS FIELDS IN THE IBUF ARRAY ARE MODIFIED
0058 214 :
0058 215 :--
0040 0058 216 .ENTRY LPASIGTBUF,^M<R6>
56 04 AC D0 005A 217 MOVL 4(AP),R6 ; GET ADDRESS OF IBUF ARRAY
005E 218
005E 219 ; REMOVE BUFFER FROM HEAD OF USER QUEUE
51 50 B6 OF 005E 220 REMQUE @IBF$Q_USRQFL(R6),R1 ; R1 GETS ADDRESS OF LINK
0C 1D 0062 221 BVS 10$ ; NO ENTRY IN QUEUE
01D4 30 0064 222 BSBW CVTADINDX ; CONVERT ADDRESS TO INDEX IN R0
66 B4 0067 223 CLRW IBF$Q_IOST(R6) ; CLEAR STATUS
02 A6 14 A6 B0 0069 224 MOVW IBF$Q_LBUF(R6),IBF$Q_IOST+2(R6) ; STORE BUFFER LENGTH
07 11 006E 225 BRB 20$

```


LPASIGTBUF - GET A BUFFER

				0070	228		
				0070	229	10\$:	; USER QUEUE IS EMPTY - RETURN REAL I/O STATUS
66	08	A6	7D	0070	230		MOVQ IBF\$Q_IOSB(R6),IBF\$Q_IOST(R6)
	50	01	CE	0074	231		MNEGL #1,R0 ; RETURN -1
				0077	232		
	02	6C	91	0077	233	20\$:	CMPB (AP),#2 ; IBUFNO SUPPLIED?
		09	1F	007A	234		BLSSU 40\$; NO
51	08	AC	D0	007C	235		MOVL 8(AP),R1 ; GET ADDRESS
		03	13	0080	236		BEQL 40\$; DEFAULTED
	61	50	D0	0082	237		MOVL R0,(R1) ; STORE INDEX
			04	0085	238	40\$:	RET ; INDEX IS IN R0 FOR FUNCTION CALL

LPA\$INXTBF - SET NEXT BUFFER

```
0086 240      .SBTTL LPA$INXTBF - SET NEXT BUFFER
0086 241      .SBTTL LPA$RMVBUF - REMOVE BUFFER
0086 242      :++
0086 243      : FUNCTIONAL DESCRIPTION:
0086 244      :
0086 245      : LPA$INXTBF CHANGES THE ORDER OF BUFFERS IN THE DEVICE QUEUE
0086 246      : BY INSERTING THE INDICATED BUFFER AT THE HEAD OF THE QUEUE.
0086 247      : NOTE THAT THE BUFFER MUST ALREADY BE ON THE DEVICE QUEUE.
0086 248      : LPA$RMVBUF REMOVES A BUFFER FROM THE DEVICE QUEUE.
0086 249      :
0086 250      : CALLING SEQUENCE:
0086 251      :
0086 252      : CALLS/G
0086 253      : THESE ROUTINES MAY ALSO BE CALLED AS FUNCTIONS
0086 254      :
0086 255      : INPUT PARAMETERS:
0086 256      :
0086 257      : IBUF(AP)          ADDRESS OF IBUF ARRAY
0086 258      : IBUFNO(AP)       ADDRESS OF LONGWORD WHICH CONTAINS INDEX
0086 259      :                   OF BUFFER TO BE REMOVED OR MADE NEXT
0086 260      : IND(AP)          ADDRESS OF LONGWORD TO RECEIVE RETURN STATUS
0086 261      :
0086 262      : IMPLICIT INPUTS:
0086 263      :
0086 264      : VARIOUS FIELDS IN THE IBUF ARRAY
0086 265      :
0086 266      : OUTPUT PARAMETERS:
0086 267      :
0086 268      : IND(AP)          ADDRESS OF LONGWORD TO RECEIVE RETURN STATUS
0086 269      :
0086 270      : IMPLICIT OUTPUTS:
0086 271      :
0086 272      : NONE
0086 273      :
0086 274      : COMPLETION CODES:
0086 275      :
0086 276      : 0      INDICATES THAT THE SPECIFIED BUFFER WAS NOT IN THE DEVICE QUEUE
0086 277      : 1      INDICATES THAT THE SPECIFIED BUFFER WAS REMOVED OR MADE NEXT
0086 278      :
0086 279      : SIDE EFFECTS:
0086 280      :
0086 281      : VARIOUS FIELDS IN THE IBUF ARRAY ARE MODIFIED
0086 282      :
0086 283      :--
0086 284      :
54 01 005C 0086 285      .ENTRY LPA$INXTBF,*M<R2,R3,R4,R6>
0086 286      MOVL #1,R4 ; INDICATES WHICH CALL
0086 287      BRB COMMON
0086 288      :
0086 289      .ENTRY LPA$RMVBUF,*M<R2,R3,R4,R6>
54 04 005C 008D 289      CLRL R4 ; INDICATES WHICH CALL
0086 290      :
0086 291      :
0086 292      :
56 04 AC D0 0091 293 COMMON: MOVL 4(AP),R6 ; GET ADDRESS OF IBUF ARRAY
0086 294      CLRL R2 ; RETURN CODE
0086 295      :
53 08 BC D0 0097 295      MOVL @8(AP),R3 ; GET BUFFER INDEX
0086 296      :
```


LPASRMVBUF - REMOVE BUFFER

```
53 68 A643 7E 009B 297      MOVAQ   IBF$Q_BFRLNKS(R6)[R3],R3; CONVERT TO ADDRESS
      00A0 298
      00A0 299
      00A0 300      ; *** TURN OFF ASTS ***
      00A9 301      $SETAST_S      #0
51 58 A6  DE 00A9 302      MOVAL   IBF$L_DEVQFL(R6),R1      ; GET ADDRESS OF DEVICE QUEUE HEAD
50 51  D0 00AD 303      MOVL    R1,R0
      00B0 304
      00B0 305 10$:      ; GET NEXT LINK IN QUEUE
50 60  D0 00B0 306      MOVL    (R0),R0
51 50  D1 00B3 307      CMPL    R0,R1
      11 13 00B6 308      BEQL    30$      ; BACK TO QUEUE HEAD YET?
      00B8 309      ; YES, BUFFER NOT ON QUEUE
53 50  D1 00B8 310      CMPL    R0,R3
      F3 12 00BB 311      BNEQ    10$      ; IS THIS THE BUFFER?
      00BD 312      ; NO
      00BD 313
50 60  OF 00BD 314      ; HAVE BUFFER. ADDRESS IS IN R0. REMOVE FROM QUEUE
      00C0 315      REMQUE   (R0),R0
      00C0 316
      00C0 317      ; IF LPASINXTBF WAS ENTRY, THEN INSERT BUFFER AT HEAD OF QUEUE.
58 A6 04 54 E9 00C0 317      BLBC    R4,20$      ; BR. IF LPASRMVBUF WAS ENTRY
      60 0E 00C3 318      INSQUE  (R0),IBF$L_DEVQFL(R6) ; INSERT AT HEAD OF DEVICE QUEUE
      00C7 319
      52 D6 00C7 320 20$: INCL    R2      ; INDICATE SUCCESS
      00C9 321
      00C9 322 30$:      ;*** TURN ON ASTS ***
      00C9 323      $SETAST_S      #1
      00D2 324
50 52  D0 00D2 325      MOVL    R2,R0
03 6C  91 00D5 326      CMPB    (AP),#3      ; RETURN STATUS
      09 1F 00D8 327      BLSSU   40$      ; IND SPECIFIED?
51 0C AC  D0 00DA 328      MOVL    12(AP),R1      ; NO
      03 13 00DE 329      BEQL    40$      ; GET ADDRESS OF IND
61 50  D0 00E0 330      MOVL    R0,(R1)      ; DEFAULTED
      04 00E3 331 40$: RET      ; STORE STATUS
```


LPAS\$WTBUF - WAIT FOR A BUFFER

```

00E4 333 .SBTTL LPAS$WTBUF - WAIT FOR A BUFFER
00E4 334 :++
00E4 335 :FUNCTIONAL DESCRIPTION:
00E4 336 :
00E4 337 : THIS ROUTINE IS CALLED BY THE USER'S PROGRAM TO GET A BUFFER
00E4 338 : FROM THE HEAD OF THE USER QUEUE. HOWEVER, UNLIKE LPAS$IGTBUF,
00E4 339 : THIS ROUTINE WAITS FOR ONE IF THE QUEUE IS EMPTY. THIS ROUTINE
00E4 340 : SHOULD BE USED IF THE USER IS SYNCHRONIZING WITH EVENT FLAGS
00E4 341 : RATHER THAN WITH A COMPLETION ROUTINE. THIS ROUTINE DOES THE FOLLOWING:
00E4 342 : 1) REMOVES A BUFFER FROM THE HEAD OF THE USER QUEUE.
00E4 343 : IF THERE IS ONE, IT RETURNS WITH THE BUFFER INDEX.
00E4 344 : 2) OTHERWISE, THE EVENT FLAG IS CLEARED.
00E4 345 : 3) ANOTHER ATTEMPT IS MADE TO REMOVE A BUFFER FROM THE
00E4 346 : HEAD OF THE USER QUEUE. IF THERE IS ONE, IT IS RETURNED.
00E4 347 : 4) OTHERWISE, THE I/O STATUS BLOCK IS CHECKED TO SEE IF THE
00E4 348 : SWEEP IS STILL IN PROGRESS. IF NOT, THE STATUS IS RETURNED.
00E4 349 : 5) OTHERWISE, WE WAIT FOR THE EVENT FLAG AND THEN GO TO STEP 1.
00E4 350 : NOTE THAT STEP 1 IS ONLY AN OPTIMIZATION: IT SAVES CLEARING THE
00E4 351 : EVENT FLAG IF A BUFFER IS ALREADY AVAILABLE.
00E4 352 :
00E4 353 : CALLING SEQUENCE:
00E4 354 :
00E4 355 : CALLS/G
00E4 356 : MAY ALSO BE CALLED AS A FUNCTION
00E4 357 :
00E4 358 : INPUT PARAMETERS:
00E4 359 :
00E4 360 : IBUF(AP) ADDRESS OF IBUF ARRAY
00E4 361 : IEFN(AP) UNUSED (PRESENT FOR COMPATIBILITY ONLY)
00E4 362 : IBUFNO(AP) ADDRESS OF A LONGWORD WHICH RECEIVES BUFFER
00E4 363 : INDEX OR -1 IF THERE ARE NO BUFFERS AND THE
00E4 364 : SWEEP HAS STOPPED.
00E4 365 :
00E4 366 : IMPLICIT INPUTS:
00E4 367 :
00E4 368 : VARIOUS FIELDS IN THE IBUF ARRAY
00E4 369 :
00E4 370 : OUTPUT PARAMETERS:
00E4 371 :
00E4 372 : IBUFNO(AP) ADDRESS OF A LONGWORD WHICH RECEIVES BUFFER
00E4 373 : INDEX OR -1 IF THERE ARE NO BUFFERS AND THE
00E4 374 : SWEEP HAS STOPPED.
00E4 375 :
00E4 376 : IMPLICIT OUTPUTS:
00E4 377 :
00E4 378 : OFFSET IBF$Q_IOST IN THE IBUF ARRAY CONTAINS ADDITIONAL STATUS
00E4 379 :
00E4 380 : COMPLETION CODES:
00E4 381 :
00E4 382 : NONE
00E4 383 :
00E4 384 : SIDE EFFECTS:
00E4 385 :
00E4 386 : VARIOUS FIELDS IN THE IBUF ARRAY ARE MODIFIED
00E4 387 :
00E4 388 :--
00E4 389

```


LPASIWTFUF - WAIT FOR A BUFFER

```
0044 00E4 390 .ENTRY LPASIWTFUF,^M<R2,R6>
      00E6 391
56 04 AC D0 00E6 392 MOVL 4(AP),R6 ; GET ADDRESS OF IBUF ARRAY
52 4E A6 9A 00EA 393 MOVZBL IBF$B_EFN(R6),R2 ; GET EVENT FLAG BEING USED
      00EE 394
      00EE 395 10$: ; REMOVE BUFFER FROM HEAD OF USER QUEUE
51 50 B6 0F 00EE 396 REMQUE @IBF$B_USRQFL(R6),R1 ; ADDRESS IN R1
      31 1C 00F2 397 BVC 70$ ; HAVE ONE
      00F4 398
      00F4 399 30$: ; USER QUEUE IS EMPTY. CLEAR EVENT FLAG
      00F4 400 $CLREF_S R2 ; CLEAR EVENT FLAG
      17 50 E9 00FD 401 BLBC -R0,40$ ; ERROR
      0100 402
      0100 403 ; TRY AGAIN TO REMOVE A BUFFER FROM HEAD OF USER QUEUE
51 50 B6 0F 0100 404 REMQUE @IBF$B_USRQFL(R6),R1 ; ADDRESS IN R1
      1F 1C 0104 405 BVC 70$ ; HAVE ONE
      0106 406
      0106 407 ; QUEUE IS STILL EMPTY. CHECK FOR ERROR OR DONE BEFORE WAITING
      0106 408 TSTW IBF$Q_IOSB(R6) ; CHECK I/O STATUS BLOCK
      11 12 0109 409 BNEQ 50$ ; ERROR OR DONE
      010B 410
      010B 411 $WAITFR_S R2 ; SWEEP IS STILL ON SO WAIT
      D7 50 E8 0114 412 BLBS -R0,10$ ; TRY AGAIN
      0117 413
      0117 414 40$: ; ERROR IN CLEARING OR WAITING FOR EVENT FLAG
      66 50 D0 0117 415 MOVL R0,IBF$Q_IOST(R6) ; RETURN ERROR CODE IN I/O STATUS
      04 11 011A 416 BRB 60$
      011C 417
      011C 418 50$: ; SWEEP FINISHED NORMALLY OR WITH ERROR. RETURN I/O STATUS BLOCK
      66 08 A6 7D 011C 419 MOVQ IBF$Q_IOSB(R6),IBF$Q_IOST(R6)
      50 01 CE 0120 420
      0A 11 0123 421 60$: MNEGL #1,R0 ; RETURN -1 INSTEAD OF INDEX
      0125 422 BRB 80$
      0125 423
      0125 424 70$: ; HAVE A BUFFER ADDRESS IN R1. CONVERT TO INDEX IN R0
      0113 30 0125 425 BSBW CVTADINDX
      66 B4 0128 426 CLRW IBF$Q_IOST(R6) ; RETURN 0 STATUS
02 A6 14 A6 B0 012A 427 MOVW IBF$B_LBUF(R6),IBF$Q_IOST+2(R6) ; RETURN BUFFER LENGTH
      012F 428
      03 6C 91 012F 429 80$: CMPB (AP),#3 ; IBUFNO SPECIFIED?
      09 1F 0132 430 BLSSU 90$ ; NO
51 0C AC D0 0134 431 MOVL 12(AP),R1 ; GET ADDRESS OF IBUFNO
      03 13 0138 432 BEQL 90$ ; DEFAULTED
      61 50 D0 013A 433 MOVL R0,(R1) ; STORE INDEX IN IBUFNO
      04 013D 434 90$: RET ; INDEX IN R0 FOR FUNCTION CALL
```


LPA\$RLSBUF - RELEASE BUFFER

```

013E 436 .SBTTL LPA$RLSBUF - RELEASE BUFFER
013E 437 :++
013E 438 : FUNCTIONAL DESCRIPTION:
013E 439 :
013E 440 : THIS ROUTINE IS CALLED BY THE USER'S PROGRAM TO RELEASE A BUFFER
013E 441 : (OR BUFFERS) ONTO THE DEVICE QUEUE. IT WORKS AS FOLLOWS:
013E 442 : 1) EACH BUFFER INDEX IS CONVERTED TO AN ADDRESS
013E 443 : 2) IF THE USW IS NOT CURRENTLY SET WITH A NEXT BUFFER INDEX
013E 444 : THEN IT IS LOADED WITH THIS BUFFER'S INDEX AND THE BUFFER
013E 445 : IS INSERTED ON THE TAIL OF THE INUSE QUEUE.
013E 446 : 3) OTHERWISE, THE BUFFER IS INSERTED ON THE TAIL OF THE
013E 447 : DEVICE QUEUE.
013E 448 : IF BUFFER OVERRUN IS NON-FATAL, THEN BUFFER 0 IS HANDLED AS A
013E 449 : SPECIAL CASE. INSTEAD OF BEING INSERTED ON THE DEVICE QUEUE,
013E 450 : A FLAG IS SET. IF THIS FLAG IS NOT SET WHEN BUFFER OVERRUN OCCURS
013E 451 : THEN THE LPA-11 HAS USED BUFFER 0 BEFORE IT HAS BEEN RELEASED.
013E 452 : THIS IS RETURNED AS A DOUBLE BUFFER OVERRUN ERROR.
013E 453 :
013E 454 : CALLING SEQUENCE:
013E 455 :
013E 456 : CALLS/G
013E 457 :
013E 458 : INPUT PARAMETERS:
013E 459 :
013E 460 : IBUF(AP) ADDRESS OF IBUF ARRAY
013E 461 : IND(AP) ADDRESS OF LONGWORD TO STORE RETURN STATUS
013E 462 : NO(AP) ADDRESS OF LONGWORD CONTAINING BUFFER INDEX
013E 463 : .
013E 464 :
013E 465 : N7(AP) ADDRESS OF LONGWORD CONTAINING BUFFER INDEX
013E 466 :
013E 467 : IMPLICIT INPUTS:
013E 468 :
013E 469 : VARIOUS FIELDS IN THE IBUF ARRAY
013E 470 :
013E 471 : OUTPUT PARAMETERS:
013E 472 :
013E 473 : IND(AP) ADDRESS OF LONGWORD TO RECEIVE RETURN STATUS
013E 474 :
013E 475 : IMPLICIT OUTPUTS:
013E 476 :
013E 477 : NONE
013E 478 :
013E 479 : COMPLETION CODES:
013E 480 :
013E 481 : 0 INDICATES ILLEGAL BUFFER NUMBER, INCORRECT # OF ARGS,
013E 482 : OR DOUBLE BUFFER OVERRUN OCCURED (THE LAST CASE CAN
013E 483 : ONLY OCCUR IF BUFFER OVERRUN IS NON-FATAL AND BUFFER 0
013E 484 : WAS RELEASED).
013E 485 : 1 INDICATES BUFFER(S) SUCCESSFULLY RELEASED
013E 486 :
013E 487 : SIDE EFFECTS:
013E 488 :
013E 489 : VARIOUS FIELDS IN THE IBUF ARRAY ARE MODIFIED
013E 490 :
013E 491 : --
013E 492 :

```


LPASRLSBUF - RELEASE BUFFER

```
005C 013E 493 .ENTRY LPASRLSBUF, ^M<R2,R3,R4,R6>
      0140 494
      52 D4 0140 495
      53 6C 9A 0142 496
56 04 AC D0 0145 497
54 0C AC DE 0149 498
      53 03 C2 014D 499
      07 53 D1 0150 500
      3F 1A 0153 501
      0155 502
      0155 503 20$: ; RELEASE NEXT BUFFER
      50 94 D0 0155 504 ; MOVL @ (R4)+, R0 ; GET INDEX
      10 12 0158 505 ; BNEQ 25$ ; NOT BUFFER 0
      015A 506
      015A 507 ; HAVE BUFFER 0. IS OVERRUN NOT FATAL?
0B 20 A6 17 E1 015A 508 ; BBC #LASV_BFROVRN, IBF$L_CMDTBL(R6), 25$ ; BR. IF NO
      015F 509
      015F 510 ; BUFFER OVERRUN IS NOT FATAL AND THIS IS BUFFER 0.
      4C A6 10 A8 015F 511 ; BISW #FLG_M_BFRORLSD, IBF$W_FLAGS(R6) ; SET BUFFER 0 RELEASED FLAG
27 4C A6 05 E5 0163 512 ; BBCC #FLG_V_DBFROVRN, IBF$W_FLAGS(R6), 50$ ; BR. IF NO DBL BFR OVRN
      2A 11 0168 513 ; BRB 60$ ; DOUBLE BUFFER OVERRUN OCCURRED
      016A 514
      016A 515 25$: ; VERIFY BUFFER # IS WITHIN RANGE
      03 00 ED 016A 516 ; CMPZV #0, #3, - ; COMPARE WITH HIGHEST BUFFER # ALLOWED
      50 22 A6 016D 517 ; IBF$L_CMDTBL+CMT$B_VBFMASK(R6), R0
      22 1F 0170 518 ; BLSSU 60$ ; SUPPLIED INDEX IS TOO HIGH - ERROR
51 68 A6 40 7E 0172 519 ; MOVAQ IBF$Q_BFRLNKS(R6)[R0], R1 ; GET ADDRESS OF BUFFER LINK
      0177 520
      0177 521 ; SHOULD WE RELEASE THIS BUFFER ONTO DEVICE QUEUE OR ONTO
      0177 522 ; INUSE QUEUE AND LOAD USW
      OF 4C A6 00 E2 0177 523 ; BBSS #FLG_V_USWSET, IBF$W_FLAGS(R6), 30$ ; BR. IF USW IS ALREADY SET
      OA 48 A6 0E E0 017C 524 ; BBS #14, IBF$W_USW(R6), 30$ ; BR. IF REQUEST IS BEING STOPPED
      0181 525
      0181 526 ; INSERT ONTO TAIL OF INUSE QUEUE AND LOAD USW
      64 B6 61 OE 0181 527 ; INSQUE (R1), @IBF$L_INUQBL(R6) ; INSERT ONTO TAIL OF INUSE QUEUE
      49 A6 50 90 0185 528 ; MOVB R0, IBF$W_USW+1(R6) ; LOAD NEXT BFR INDEX AND CLEAR DONE BIT
      04 11 0189 529 ; BRB 50$
      018B 530
      018B 531 30$: ; INSERT BUFFER ONTO TAIL OF DEVICE QUEUE
      5C B6 61 OE 018B 532 ; INSQUE (R1), @IBF$L_DEVQBL(R6)
      018F 533
      C3 53 F4 018F 534 50$: ; SOBGEQ R3, 20$ ; DO NEXT ONE
      52 D6 0192 535 ; INCL R2 ; INDICATE SUCCESS
      0194 536
      50 08 AC D0 0194 537 60$: ; MOVL 8(AP), R0 ; GET ADDRESS OF IND
      03 13 0198 538 ; BEQL 70$ ; DEFAULTED
      60 52 D0 019A 539 ; MOVL R2, (R0) ; STORE RETURN STATUS
      04 019D 540 70$: ; RET
```

LPASSBFRAST - BUFFER FULL/EMPTY AST HAND

```
019E 542 .SBTTL LPASSBFRAST - BUFFER FULL/EMPTY AST HANDLER
019E 543 :++
019E 544 : FUNCTIONAL DESCRIPTION:
019E 545 :
019E 546 : THIS ROUTINE IS THE AST HANDLER FOR NORMAL BUFFER FULL OR
019E 547 : EMPTY ASTS. PRIMARILY IT DOES THE FOLLOWING:
019E 548 : 1) REMOVE A BUFFER FROM HEAD OF DEVICE QUEUE
019E 549 : 2) LOAD THAT BUFFER'S INDEX INTO THE USW
019E 550 : 3) INSERT THAT BUFFER INTO TAIL OF INUSE QUEUE
019E 551 : 4) REMOVE A BUFFER FROM HEAD OF INUSE QUEUE
019E 552 : 5) INSERT THAT BUFFER INTO TAIL OF USER QUEUE
019E 553 :
019E 554 : CALLING SEQUENCE:
019E 555 :
019E 556 : CALLS/G
019E 557 :
019E 558 : INPUT PARAMETERS:
019E 559 :
019E 560 : IBUF(AP) ADDRESS OF IBUF ARRAY (AST PARAMETER)
019E 561 : 8(AP) SAVED R0 (UNUSED)
019E 562 : 12(AP) SAVED R1 (UNUSED)
019E 563 : 16(AP) SAVED PC (UNUSED)
019E 564 : 20(AP) SAVED PSL (UNUSED)
019E 565 :
019E 566 : IMPLICIT INPUTS:
019E 567 :
019E 568 : VARIOUS FIELDS IN THE IBUF ARRAY
019E 569 :
019E 570 : OUTPUT PARAMETERS:
019E 571 :
019E 572 : NONE
019E 573 :
019E 574 : IMPLICIT OUTPUTS:
019E 575 :
019E 576 : NONE
019E 577 :
019E 578 : COMPLETION CODES:
019E 579 :
019E 580 : NONE
019E 581 :
019E 582 : SIDE EFFECTS:
019E 583 :
019E 584 : VARIOUS FIELDS IN THE IBUF ARRAY ARE MODIFIED
019E 585 :
019E 586 : --
019E 587 :
0040 019E 588 .ENTRY LPASSBFRAST,^M<R6>
01A0 019E 589
56 04 AC D0 01A0 590 MOVL 4(AP),R6 ; GET ADDRESS OF IBUF
01A4 591
27 48 A6 0E E0 01A4 592 BBS #14,IBF$W_USW(R6),40$ ; BR. IF STOP BIT SET IN USW
0B 4C A6 02 E1 01A9 593 BBC #FLG V CNTBFRS,IBF$W_FLAGS(R6),10$ ; BR. IF NOT COUNTING BUFFERS
07 18 A6 F5 01AE 594 SOBGTR IBF$C_NBUF(R6),10$ ; BR. IF WE HAVEN'T FINISHED NBUF BFRS
01B2 595
01B2 596 ; NBUF BUFFERS HAVE BEEN FILLED OR EMPTIED. STOP SWEEP
49 A6 40 8F 90 01B2 597 MOVB #^X40,IBF$W_USW+1(R6) ; SET STOP BIT IN USW
17 11 01B7 598 BRB 40$
```



```
LPASSBFRST - BUFFER FULL/EMPTY AST HAND

      01B9 599
51 58 B6 0F 01B9 600 10$: ; REMOVE BFR FROM DEV. Q, LOAD INTO USW, AND INSERT INTO INUSE Q.
      01B9 601 REMQUE @IBF$L_DEVQFL(R6),R1 ; REMOVE FROM HEAD OF DEVICE QUEUE
      01BD 602 BVC 20$ ; HAVE ONE
4C A6 01 AA 01BF 603 BICW #FLG_M_USWSET,IBF$W_FLAGS(R6) ; NONE IN Q. CLEAR BIT FOR RLSBUF
      01C3 604 BRB 40$
      01C5 605 20$: BSBW CVTADINDX ; CONVERT ADDRESS IN R1 TO INDEX IN R0
49 A6 50 90 01C8 606 MOVB R0,IBF$W_USW+1(R6) ; LOAD USW (THIS ALSO CLEARS DONE BIT)
64 B6 61 0E 01CC 607 INSQUE (R1),@IBF$L_INUQBL(R6) ; INSERT IN TAIL OF INUSE QUEUE
      01D0 608
      01D0 609 40$: ; IF THIS IS THE FIRST AST (REQUEST STARTED) DON'T PUT A BUFFER
      01D0 610 ; ON THE USER QUEUE
03 4C A6 01 E3 01D0 611 BBCS #FLG_V_SWPSTRTD,IBF$W_FLAGS(R6),80$ ; BR. IF FIRST TIME THROUGH
      01D5 612
      01D5 613 ; NOW REMOVE FROM HEAD OF INUSE QUEUE, INSERT ON TAIL OF USER QUEUE
      01D5 614 ; AND CALL USER'S COMPLETION ROUTINE IF ONE WAS SPECIFIED.
      01D5 615 BSBW GIVUSRBFR ; DOES IT ALL
      01D8 616
      01D8 617 80$: RET
006F 30
      04 01D8
```

LPA\$SOVRAS - BFR OVER/UNDERRUN AST HAND

```
01D9 619 .SBTTL LPA$SOVRAS - BFR OVER/UNDERRUN AST HANDLER
01D9 620 :++
01D9 621 : FUNCTIONAL DESCRIPTION:
01D9 622 :
01D9 623 : THIS ROUTINE IS THE AST HANDLER FOR BUFFER OVER/UNDERRUN ASTS.
01D9 624 : PRIMARILY, IT DOES THE FOLLOWING:
01D9 625 : 1) REMOVE A BUFFER FROM HEAD OF INUSE QUEUE
01D9 626 : 2) INSERT THAT BUFFER INTO TAIL OF USER QUEUE
01D9 627 : 3) IF BUFFER 0 HAS BEEN RELEASED, THEN IT IS INSERTED INTO
01D9 628 : THE HEAD (!) OF THE INUSE QUEUE. OTHERWISE, THE
01D9 629 : DOUBLE BUFFER OVERRUN BIT IS SET.
01D9 630 :
01D9 631 : CALLING SEQUENCE:
01D9 632 :
01D9 633 : CALLS/G
01D9 634 :
01D9 635 : INPUT PARAMETERS:
01D9 636 :
01D9 637 : IBUF(AP) ADDRESS OF IBUF ARRAY (AST PARAMETER)
01D9 638 : 8(AP) SAVED R0 (UNUSED)
01D9 639 : 12(AP) SAVED R1 (UNUSED)
01D9 640 : 16(AP) SAVED PC (UNUSED)
01D9 641 : 20(AP) SAVED PSL (UNUSED)
01D9 642 :
01D9 643 : IMPLICIT INPUTS:
01D9 644 :
01D9 645 : VARIOUS FIELDS IN THE IBUF ARRAY
01D9 646 :
01D9 647 : OUTPUT PARAMETERS:
01D9 648 :
01D9 649 : NONE
01D9 650 :
01D9 651 : IMPLICIT OUTPUTS:
01D9 652 :
01D9 653 : NONE
01D9 654 :
01D9 655 : COMPLETION CODES:
01D9 656 :
01D9 657 : NONE
01D9 658 :
01D9 659 : SIDE EFFECTS:
01D9 660 :
01D9 661 : VARIOUS FIELDS IN THE IBUF ARRAY ARE MODIFIED
01D9 662 : --
01D9 663 :
0040 01D9 664 .ENTRY LPA$SOVRAS,^M<R6>
01DB 665
56 04 AC D0 01DB 666 MOVL 4(AP),R6 ; GET ADDRESS OF IBUF
01DF 667
01DF 668 ; REMOVE HEAD OF INUSE QUEUE, INSERT ON TAIL OF USER QUEUE,
01DF 669 ; AND CALL USER'S COMPLETION ROUTINE IF ONE WAS SPECIFIED.
0065 30 01DF 670 BSBW GIVUSRBF ; DOES IT ALL
01E2 671
01E2 672 ; MAKE SURE BUFFER 0 HAS BEEN RELEASED.
06 4C A6 04 E4 01E2 673 BBSC #FLG_V_BFRORLSD,IBF$W_FLAGS(R6),10$ ; BR. IF IT HAS BEEN
4C A6 20 A8 01E7 674 BBSW #FLG_M_DBFROVRN,IBF$W_FLAGS(R6) ; DBL BFR OVRN OCCURRED
05 11 01EB 675 BRB 20$
```


60 A6	68 A6	0E	01ED	676		
			01ED	677	10\$:	: INSERT BUFFER 0 INTO HEAD OF INUSE QUEUE
			01ED	678		INSQUE IBF\$Q_BFRLNKS(R6),IBF\$L_INUQFL(R6)
			01F2	679		
09 4C A6	02	E1	01F2	680	20\$:	BBC #FLG V CNTBFRS,IBF\$W_FLAGS(R6),30\$; BR. IF NOT COUNTING BFRS
05 18 A6		F5	01F7	681		SOBGTR IBF\$L_NBUF(R6),30\$; BR. IF WE HAVEN'T FINISHED NBUF BFRS
49 A6	40 8F	90	01FB	682		MOVB #^X40,IBF\$W_USW+1(R6) ; SET STOP BIT IN USW
			0200	683		
		04	0200	684	30\$:	RET

```
0201 686 .SBTTL LPAS$CMPLTAST - QIO COMPLETE AST HANDLER
0201 687 :++
0201 688 : FUNCTIONAL DESCRIPTION:
0201 689 :
0201 690 : THIS ROUTINE IS THE QIO COMPLETE AST HANDLER. PRIMARILY, IT DOES
0201 691 : THE FOLLOWING:
0201 692 : 1) UNDER CERTAIN CONDITIONS THERE IS AN EXTRA BUFFER TO BE PLACED
0201 693 : ON THE USER QUEUE. IF THERE IS, A BUFFER IS REMOVED FROM THE
0201 694 : HEAD OF THE INUSE QUEUE AND INSERTED ON THE TAIL OF THE USER QUEUE.
0201 695 : 2) IF THE USER SPECIFIED A COMPLETION ROUTINE, IT IS CALLED.
0201 696 : 3) THE CHANNEL IS DEASSIGNED.
0201 697 :
0201 698 : CALLING SEQUENCE:
0201 699 :
0201 700 : CALLS/G
0201 701 :
0201 702 : INPUT PARAMETERS:
0201 703 :
0201 704 : IBUF(AP) ADDRESS OF IBUF ARRAY (AST PARAMETER)
0201 705 : 8(AP) SAVED R0 (UNUSED)
0201 706 : 12(AP) SAVED R1 (UNUSED)
0201 707 : 16(AP) SAVED PC (UNUSED)
0201 708 : 20(AP) SAVED PSL (UNUSED)
0201 709 :
0201 710 : IMPLICIT INPUTS:
0201 711 :
0201 712 : VARIOUS FIELDS IN THE IBUF ARRAY
0201 713 :
0201 714 : OUTPUT PARAMETERS:
0201 715 :
0201 716 : NONE
0201 717 :
0201 718 : IMPLICIT OUTPUTS:
0201 719 :
0201 720 : NONE
0201 721 :
0201 722 : COMPLETION CODES:
0201 723 :
0201 724 : NONE
0201 725 :
0201 726 : SIDE EFFECTS:
0201 727 :
0201 728 : VARIOUS FIELDS IN THE IBUF ARRAY ARE MODIFIED
0201 729 :
0201 730 :--
0201 731 :
0040 0201 732 .ENTRY LPAS$CMPLTAST,^M<R6>
56 04 AC D0 0203 733
0207 734 MOVL 4(AP),R6 ; GET ADDRESS OF IBUF
0207 735
0207 736 ; WE HAVE AN EXTRA BUFFER IF THE STATUS IS NORMAL, BUFFER OVERRUN,
0207 737 ; OR NON-FATAL ERROR COUNT EXCEEDED (SAME AS BUFFER OVERRUN).
00' 08 A6 B1 0207 738 CMPW IBF$Q_IOSB(R6),S^#SS$_NORMAL ; NORMAL STATUS?
0208 739 BEQL 20$ ; YES
0000'8F 08 A6 B1 020D 740 CMPW IBF$Q_IOSB(R6),#SS$_DEVREQERR ; REQUEST ERROR?
A3 8F 0D A6 91 0213 741 BNEQ 40$ ; NO, SO IT CAN'T BE EITHER OTHER ERROR
0215 742 CMPB IBF$Q_IOSB+5(R6),#BFROVRN ; BUFFER OVER/UNDERRUN?
```



```

AO 8F  OD 07 13 021A 743      BEQL 20$      ; YES
          03 12 021C 744      CMPB IBF$Q_IOSB+5(R6),#NFECEX ; NON-FATAL ERROR COUNT EXC.?
          03 12 0221 745      BNEQ 40$      ; NO
          03 12 0223 746
          03 12 0223 747 20$:  ; HAVE AN EXTRA BUFFER TO PLACE ON USER QUEUE. REMOVE FROM HEAD
          03 12 0223 748      ; OF INUSE QUEUE, INSERT ON TAIL OF USER QUEUE, AND CALL USER'S
          03 12 0223 749      ; COMPLETION ROUTINE IF ONE WAS SPECIFIED.
          0021 30 0223 750      BSBW GIVUSRBFR ; DOES IT ALL
          03 12 0226 751
          03 12 0226 752 40$:  ; NOW CALL USER'S COMPLETION ROUTINE (POSSIBLY FOR THE SECOND TIME)
          03 12 0226 753      ; IF ONE WAS SPECIFIED.
          50 10 A6 D0 0226 754      MOVL IBF$L_COMPLADDR(R6),R0 ; GET ADDRESS OF ROUTINE
          03 13 022A 755      BEQL 50$      ; NONE SPECIFIED
          60 00 FB 022C 756      CALLS #0,(R0) ; CALL IT
          03 12 022F 757
          03 12 022F 758 50$:  ; DEASSIGN CHANNEL
          03 12 022F 759      $DASSGN_S IBF$W_CHAN(R6)
          04 023A 760
          04 023A 761      RET

```

LP
Sy
SS
CH
CH
DI
DI
IO
IO
IO
IO
LP
MB
MB
MB
MB
MB
RM
SM
SM
SM
SS
SY
SY
SY
SY
WR

PS
--

\$A
_L

Ph
--
In
Co
Pa
Sy
Pa
Sy
Ps
Cr
As

Th
24

CVTADINDX - CONVERT ADDRESS TO INDEX

```

023B 763 .SBTTL CVTADINDX - CONVERT ADDRESS TO INDEX
023B 764 :++
023B 765 : FUNCTIONAL DESCRIPTION:
023B 766 :
023B 767 : THIS ROUTINE CONVERTS A BUFFER LINK ADDRESS TO A BUFFER INDEX
023B 768 :
023B 769 : CALLING SEQUENCE:
023B 770 :
023B 771 : BSBW/B
023B 772 :
023B 773 : INPUT PARAMETERS:
023B 774 :
023B 775 : R1 ADDRESS OF BUFFER LINK
023B 776 : R6 ADDRESS OF IBUF ARRAY
023B 777 :
023B 778 : IMPLICIT INPUTS:
023B 779 :
023B 780 : NONE
023B 781 :
023B 782 : OUTPUT PARAMETERS:
023B 783 :
023B 784 : R0 CONTAINS BUFFER INDEX
023B 785 :
023B 786 : IMPLICIT OUTPUTS:
023B 787 :
023B 788 : NONE
023B 789 :
023B 790 : COMPLETION CODES:
023B 791 :
023B 792 : NONE
023B 793 :
023B 794 : SIDE EFFECTS:
023B 795 :
023B 796 : NONE
023B 797 :
023B 798 : --
023B 799 :

```

```

50 50 68 A6 9E 023B 800 CVTADINDX:
50 51 50 C3 023F 801 MOVAB IBF$Q_BFRLNKS(R6),R0 ; GET ADDRESS OF START OF BUFFER LINKS
50 50 08 C6 0243 802 SUBL3 R0,R1,R0 ; SUBTRACT THAT FROM GIVEN BFR LINK ADDR
05 0246 803 DIVL #8,R0 ; DIVIDE BY 8 TO CONVERT TO INDEX
05 0246 804 RSB

```


GIVUSRBFR - GIVE USER BUFFER

```
0247 806 .SBTTL GIVUSRBFR - GIVE USER BUFFER
0247 807 :++
0247 808 : FUNCTIONAL DESCRIPTION:
0247 809 :
0247 810 : THIS ROUTINE REMOVES A BUFFER FROM THE HEAD OF THE INUSE QUEUE.
0247 811 : INSERTS IT ON THE TAIL OF THE USER QUEUE, AND CALLS THE USER'S
0247 812 : COMPLETION ROUTINE, IF ONE WAS SPECIFIED.
0247 813 :
0247 814 : CALLING SEQUENCE:
0247 815 :
0247 816 : BSBW/B
0247 817 :
0247 818 : INPUT PARAMETERS:
0247 819 :
0247 820 : R6 ADDRESS OF IBUF ARRAY
0247 821 :
0247 822 : IMPLICIT INPUTS:
0247 823 :
0247 824 : VARIOUS FIELDS IN THE IBUF ARRAY
0247 825 :
0247 826 : OUTPUT PARAMETERS:
0247 827 :
0247 828 : NONE
0247 829 :
0247 830 : IMPLICIT OUTPUTS:
0247 831 :
0247 832 : NONE
0247 833 :
0247 834 : COMPLETION CODES:
0247 835 :
0247 836 : NONE
0247 837 :
0247 838 : SIDE EFFECTS:
0247 839 :
0247 840 : R0,R1 ARE NOT PRESERVED
0247 841 : VARIOUS FIELDS IN THE IBUF ARRAY ARE MODIFIED
0247 842 :
0247 843 :--
0247 844 :
0247 845 GIVUSRBFR:
0247 846 : REMOVE BUFFER FROM HEAD OF INUSE QUEUE
50 60 B6 0F 0247 847 REMQUE @IBF$L_INUGFL(R6),R0 : R0 CONTAINS ADDRESS
0247 848 BVS 20$ : QUEUE IS EMPTY - SHOULD NEVER HAPPEN!
0247 849
0247 850 : INSERT BUFFER ON TAIL OF USER QUEUE
54 B6 60 0E 0247 851 INSQUE (R0),@IBF$L_USRQBL(R6)
0247 852
0247 853 : CALL USER'S COMPLETION ROUTINE IF ONE WAS SPECIFIED
50 10 A6 D0 0247 854 MOVL IBF$L_COMPLADDR(R6),R0 : GET ADDRESS OF COMPLETION ROUTINE
0247 855 BEQL 20$ : NONE SPECIFIED
60 00 03 13 0247 856 CALLS #0,(R0) : CALL IT
0247 857 20$: RSB
0247 858
0247 859
0247 860
0247 861 LPASBFEND:: : END OF BUFFER ROUTINES (MAY BE USED
0247 862 : TO LOCK THESE ROUTINES INTO THE
```


LPASBUFFER
V04-000

GIVUSRBFR - GIVE USER BUFFER

0258 863
0258 864
0258 865
0258 866

.END

C 9

16-SEP-1984 01:44:42 VAX/VMS Macro V04-00
5-SEP-1984 01:32:11 [IOSUP.SRC]LABUFFER.MAR;1

Page 21
(12)

; PROCESS'S WORKING SET).

LP
Tab

LPASBUFFER
Symbol table

BFROVRN	=	000000A3		
CLKEVFLG	=	00000017		
CMT\$B_EVMRKN		00000021		
CMT\$B_ICHN		0000001A		
CMT\$B_INC		0000001B		
CMT\$B-STWRDN		00000020		
CMT\$B-VBFRMASK		00000002		
CMT\$B-BFRADDR		0000000C		
CMT\$B-BFRLEN		00000008		
CMT\$B-RCLADDR		00000014		
CMT\$B-RCLLEN		00000010		
CMT\$B-USWADDR		00000004		
CMT\$W-DELAY		00000018		
CMT\$W-DWELL		0000001E		
CMT\$W-EVMRKM		00000024		
CMT\$W-MODE		00000000		
CMT\$W-NCHN		0000001C		
CMT\$W-STWRDM		00000022		
COMMON		00000091	R	02
CVTADINDX		0000023B	R	02
DEFEVFLG	=	00000016		
FLG_M-BFRORLSD	=	00000010		
FLG_M-DBFROVRN	=	00000020		
FLG_M-USWSET	=	00000001		
FLG_V-BFRORLSD	=	00000004		
FLG_V-CNTBFRS	=	00000002		
FLG_V-DBFROVRN	=	00000005		
FLG_V-SWPSTRD	=	00000001		
FLG_V-USWSET	=	00000000		
FOLCOWQ		00000043	R	02
GIVUSRBF		00000247	R	02
IBF\$B-EFN		0000004E		
IBF\$B-CMDTBL		00000020		
IBF\$B-COMPLADDR		00000010		
IBF\$B-DEVQBL		0000005C		
IBF\$B-DEVQFL		00000058		
IBF\$B-INUQBL		00000064		
IBF\$B-INUQFL		00000060		
IBF\$B-LAMSKB		0000001C		
IBF\$B-LBUF		00000014		
IBF\$B-NBUF		00000018		
IBF\$B-USRQBL		00000054		
IBF\$B-USRQFL		00000050		
IBF\$Q-BFRLNKS		00000068		
IBF\$Q-IOB		00000008		
IBF\$Q-IOST		00000000		
IBF\$W-CHAN		0000004A		
IBF\$W-FLAGS		0000004C		
IBF\$W-USW		00000048		
INITCODE	=	00001234		
LA\$V-BFROVRN	=	00000017		
LPAS\$BFRST		0000019E	RG	02
LPAS\$CMPLTAST		00000201	RG	02
LPAS\$OVRST		000001D9	RG	02
LPAS\$BFEND		0000025B	RG	02
LPAS\$BFSTART		00000000	RG	02
LPAS\$IBFSTS		00000000	RG	02

LPAS\$IGTBUF	00000058	RG	02
LPAS\$INXTBF	00000086	RG	02
LPAS\$IWTBUF	000000E4	RG	02
LPAS\$RLSBUF	0000013E	RG	02
LPAS\$RMVBUF	0000008D	RG	02
NFECEX	= 000000A0		
SIZ...	= 00000001		
SS\$-DEVREQERR	*****	X	02
SS\$-NORMAL	*****	X	02
SYSS\$CLREF	*****	GX	02
SYSS\$DASSGN	*****	GX	02
SYSS\$SETAST	*****	GX	02
SYSS\$WAITFR	*****	GX	02

D 9

16-SEP-1984 01:44:42
5-SEP-1984 01:32:11

VAX/VMS Macro V04-00
[IOSUP.SRC]LABUFFER.MAR;1

Page 22
(12)

LP
V04

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$AB\$\$	000000A8 (168.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_LPA\$CODE	0000025B (603.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	34	00:00:00.06	00:00:00.93
Command processing	139	00:00:00.63	00:00:02.32
Pass 1	155	00:00:03.25	00:00:07.09
Symbol table sort	0	00:00:00.15	00:00:00.27
Pass 2	150	00:00:01.86	00:00:03.63
Symbol table output	7	00:00:00.07	00:00:00.28
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	490	00:00:06.06	00:00:14.57

The working set limit was 1200 pages.
18381 bytes (36 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 111 non-local and 35 local symbols.
995 source lines were read in Pass 1, producing 37 object records in Pass 2.
15 pages of virtual memory were used to define 14 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	9

121 GETS were required to define 9 macros.

There were no errors, warnings or information messages.

MACRO/DISABLE=TRACE/LIS=LISS:LABUFFER/OBJ=OBJ\$:LABUFFER MSRC\$:LADEF/UPDATE=(ENH\$:LADEF)+MSRC\$:LABUFFER/UPDATE=(ENH\$:LABUFFER)

0190 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

