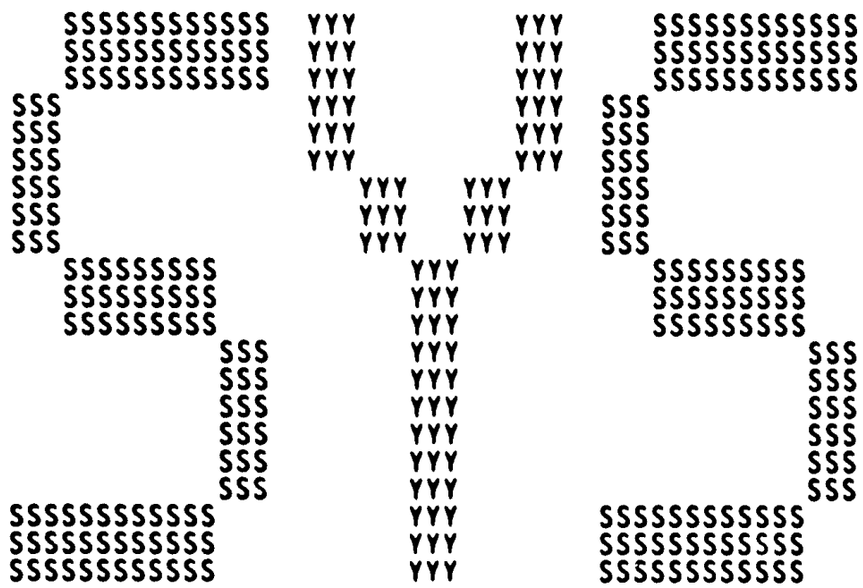

```
  AAAAAA      SSSSSSSS  TTTTTTTTTT  DDDDDDDD    EEEEEEEEEE  LL
  AAAAAA      SSSSSSSS  TTTTTTTTTT  DDDDDDDD    EEEEEEEEEE  LL
AA      AA  SS            TT      DD      DD  EE          LL
AA      AA  SS            TT      DD      DD  EE          LL
AA      AA  SS            TT      DD      DD  EE          LL
AA      AA  SS            TT      DD      DD  EE          LL
AA      AA    SSSSSS      TT      DD      DD  EEEEEEEE    LL
AA      AA    SSSSSS      TT      DD      DD  EEEEEEEE    LL
AAAAAAAAAA          SS    TT      DD      DD  EE          LL
AAAAAAAAAA          SS    TT      DD      DD  EE          LL
AA      AA          SS    TT      DD      DD  EE          LL              ....
AA      AA          SS    TT      DD      DD  EE          LL              ....
AA      AA  SSSSSSSS      TT      DDDDDDDD    EEEEEEEEEE  LLLLLLLLLL      ....
AA      AA  SSSSSSSS      TT      DDDDDDDD    EEEEEEEEEE  LLLLLLLLLL      ....


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

Table of contents

```
0000     1          .TITLE  ASTDEL - AST ENQUEUE AND DELIVERY
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  ; FACILITY: EXECUTIVE, SCHEDULER
0000    30  ;
0000    31  ; ABSTRACT:
0000    32  ;       ASTDEL CONTAINS THE AST DELIVERY INTERRUPT SERVICE ROUTINE AND THE
0000    33  ;       ASSOCIATED SUBROUTINES SCH$QAST AND SCH$NEWLVL.  THESE ROUTINES
0000    34  ;       IMPLEMENT THE PRIMITIVE AST QUEUEING AND DELIVERY MECHANISMS.
0000    35  ;
0000    36  ; ENVIRONMENT:
0000    37  ;       MODE = KERNEL
0000    38  ;--
0000    39  ;
0000    40  ;       .PAGE
0000    41         .SBTTL  HISTORY                 ; DETAILED
0000    42  ;
0000    43  ; AUTHOR:       R. HUSTVEDT      CREATION DATE: 1-SEP-76
0000    44  ;
0000    45  ; MODIFIED BY:
0000    46  ;
0000    47  ;       V03-003 SRB0137         Steve Beckhardt         12-Jul-1984
0000    48  ;               Fixed bug where ACBs queued to non-existent processes
0000    49  ;               are deallocated without checking the NODELETE bit.
0000    50  ;
0000    51  ;       V03-002 WMC0001         Wayne Cardoza           29-Feb-1984
0000    52  ;               Optimize queuing to non-resident process.
0000    53  ;
0000    54  ;       V03-001 ACG0341         Andrew C. Goldstein,    10-Jun-1983  18:59
0000    55  ;               Check and correct stack pointer before AST delivery
0000    56  ;               to mask 780 microcode bug in RET instruction.
0000    57  ;
```

0000 58

```
                     0000     60          .SBTTL  DECLARATIONS
                     0000     61
                     0000     62 ;
                     0000     63 ; INCLUDE FILES:
                     0000     64 ;
                     0000     65          $ACBDEF                         ; AST CONTROL BLOCK DEFINITIONS
                     0000     66          $IPLDEF                         ; IPL DEFINITIONS
                     0000     67          $PCBDEF                         ; PCB DEFINITIONS
                     0000     68          $PHDDEF                         ; PHD DEFINITIONS
                     0000     69          $PRDEF                          ; PROCESSOR REGISTER DEFINITIONS
                     0000     70          $PRIDEF                         ; PRIORITY INCREMENT CLASS DEFS
                     0000     71          $PSLDEF                         ; PSL FIELD DEFINITIONS
                     0000     72          $SSDEF                          ; STATUS CODE DEFINITIONS
                     0000     73
                     0000     74 ;
                     0000     75 ; EQUATED SYMBOLS:
                     0000     76 ;
00000000             0000     77 ASTEXIT=0                               ; AST EXIT CHANGE MODE CODE
                     0000     78 ;
                     0000     79 ; OWN STORAGE:
                     0000     80 ;
                00000000     81          .PSECT  A$EXENONPAGED,LONG
                     0000     82
```

```
                     0000     84          .SBTTL  SCH$ASTDEL - AST DELIVERY INTERRUPT HANDLER
                     0000     85
                     0000     86 ;++
                     0000     87 ; FUNCTIONAL DESCRIPTION:
                     0000     88 ;       SCH$ASTDEL RECEIVES THE AST DELIVERY INTERRUPT (IPL - 2) WHICH
                     0000     89 ;       IS INITIATED BY AN REI INSTRUCTION DETECTING ASTLVL LESS THAN
                     0000     90 ;       OR EQUAL TO PSL<CURRENT_MODE>.  THE HEAD OF THE AST QUEUE
                     0000     91 ;       FOR THE CURRENT PROCESS IS REMOVED AND PROCESSED.  SPECIAL
                     0000     92 ;       KERNEL MODE ASTS ARE PROCESSED WITH IPL REMAINING AT IPL 2.
                     0000     93 ;       NORMAL ASTS ARE DELIVERED BY PUSHING THE AST INFORMATION ON
                     0000     94 ;       THE STACK OF THE MODE RECEIVING THE AST AND THE AST ACTIVE
                     0000     95 ;       BIT FOR THAT MODE IS SET TO PREVENT SUBSEQUENT ASTS UNTIL THE
                     0000     96 ;       CURRENT ONE FOR THAT MODE HAS BEEN PROCESSED.
                     0000     97 ;       SPURIOUS AST INTERRUPTS WILL BE DETECTED AND IGNORED.
                     0000     98 ;
                     0000     99 ;       THIS ROUTINE CONTAINS TWO HOOKS FOR LOADABLE MULTI-PROCESSING
                     0000    100 ;       CODE.  THE FIRST HOOK, MPH$ASTDELHK, IS REPLACED BY A BRANCH
                     0000    101 ;       TO THE LOADABLE CODE, WHILE THE SECOND HOOK, MPH$ASTDELCONT, IS
                     0000    102 ;       THE ENTRY POINT AT WHICH THE LOADABLE CODE REJOINS THE COMMON
                     0000    103 ;       LINE OF EXECUTION.
                     0000    104 ;
                     0000    105 ;
                     0000    106 ; CALLING SEQUENCE:
                     0000    107 ;       IPL - 2  INTERRUPT
                     0000    108 ;
                     0000    109 ;
                     0000    110 ; INPUT PARAMETERS:
                     0000    111 ;       00(SP) = PC AT AST DELIVERY INTERRUPT
                     0000    112 ;       04(SP) = PSL AT AST DELIVERY INTERRUPT
                     0000    113 ;
                     0000    114 ; IMPLICIT INPUTS:
                     0000    115 ;       PCB OF CURRENT PROCESS LOCATED VIA SCH$GL_CURPCB
                     0000    116 ;       AST CONTROL BLOCK AT HEAD OF AST QUEUE FOR PROCESS
                     0000    117 ;
                     0000    118 ; OUTPUT PARAMETERS:
                     0000    119 ;       NONE
                     0000    120 ;
                     0000    121 ; COMPLETION CODES:
                     0000    122 ;       NONE
                     0000    123 ;--
                     0000    124
                     0000    125          .ALIGN  LONG                       ; INTERRUPT ROUTINES ON LW BOUND
                     0000    126 SCH$ASTDEL::                                ; AST DELIVERY INTERRUPT HANDLER
          55    DD   0000    127          PUSHL   R5                         ; SAVE R0-R5
          54    DD   0002    128          PUSHL   R4                         ; ONE REGISTER AT A TIME FOR
          53    DD   0004    129          PUSHL   R3                         ; SPEED AT THE
          52    DD   0006    130          PUSHL   R2                         ; EXPENSE OF SPACE AND
          51    DD   0008    131          PUSHL   R1                         ; CLARITY
          50    DD   000A    132          PUSHL   R0                         ; (PUSHLS ARE FASTER THAN MOVQ OR PUSHR)
    54  0000'CF   D0   000C    133 GETNEXT:MOVL    W^SCH$GL_CURPCB,R4         ; GET POINTER TO CURRENT PCB
                     0011    134          SETIPL  #IPL$_SYNCH                ; BLOCK SYSTEM EVENTS
                     0014    135 MPH$ASTDELHK::                              ; MULTI-PROCESSING CHECK HOOKS IN HERE
    55    10 B4   0F   0014    136          REMQUE  @PCB$L_ASTQFL(R4),R5       ; AND REMOVE HEAD OF QUEUE
          14   1D   0018    137          BVS     QEMPTY                     ; EXIT IF QUEUE EMPTY
                     001A    138 MPH$ASTDELCONT::                            ; MULTI-PROCESSING CHECK RETURNS TO HERE
28 0B A5   07   E5   001A    139          BBCC    #ACB$V_KAST,ACB$B_RMOD(R5),NORM ; BR IF NORMAL AST
```

```
                     001F   141          .SBTTL  KAST - SPECIAL KERNEL AST DISPATCHING
                     001F   142 ;
                     001F   143 ; AST LEVEL WILL BE LEFT AT 0 (KERNEL) WHILE PROCESSING THE SPECIAL KERNEL
                     001F   144 ; AST.  IT WILL BE CORRECTED EVENTUALLY BEFORE IPL IS DROPPED BELOW IPL$_ASTDEL
                     001F   145 ; BY REPEATED TRIPS TO GETNEXT.
                     001F   146 ;
                     001F   147 ; THE KERNEL AST ROUTINE IS ENTERED VIA A JSB TO THE SPECIFIED
                     001F   148 ; ADDRESS WITH IPL=2 AND THE POINTER TO THE AST CONTROL BLOCK
                     001F   149 ; IN R5.  IT IS THE RESPONSIBILITY OF THE KERNEL AST ROUTINE
                     001F   150 ; TO PROPERLY RELEASE OR OTHERWISE DISPOSE OF THE AST CONTROL
                     001F   151 ; BLOCK.  THE PCB BASE ADDRESS IS IN R4.
                     001F   152 ;
                     001F   153 ; REGISTERS R0-R5 HAVE BEEN PRESERVED AND ARE AVAILABLE FOR
                     001F   154 ; USE BY THE AST ROUTINE.
                     001F   155 ;
                     001F   156 ;
                     001F   157 ; SINCE KAST ROUTINES OFTEN QUEUE NORMAL ASTS, QIO FOR EXAMPLE, ATTEMPT
                     001F   158 ; TO DELIVER FURTHER ASTS WITHOUT INCURRING REDUNDANT EXIT AND ENTRY COSTS.
                     001F   159 ;
                     001F   160
EA AF    9F          001F   161          PUSHAB  GETNEXT                 ; SET RETURN ADDRESS TO CONTINUE WITH Q
                     0022   162          SETIPL  #IPL$_ASTDEL            ; DROP IPL TO PERMIT SYSTEM EVENTS
18 B5    17          0025   163          JMP     @ACB$L_KAST(R5)         ; DO KERNEL MODE AST
                     0028   164
                     0028   165 ;
                     0028   166 ; THIS UNUSUAL CALLING SEQUENCE IS TO MINIMIZE THE NUMBER OF TAKEN BRANCHES
                     0028   167 ; AND IS EQUIVALENT TO:
                     0028   168 ;          JSB     @ACB$L_KAST(R5)
                     0028   169 ;          BRB     GETNEXT
                     0028   170 ;
```

```
                     0028   172          .SBTTL  ASTDEL EXITS
                     0028   173 ;
                     0028   174 ; IF THE ASTMODE IS DISABLED OR ACTIVE, THEN SET ASTLVL TO PREVENT
                     0028   175 ; FURTHER INTERRUPTS.  THERE IS AN ASSUMPTION (AND HAS BEEN FOREVER)
                     0028   176 ; THAT AN INNER ACCESS MODE WILL NOT EXIT TO AN OUTER ACCESS MODE
                     0028   177 ; WITH EITHER AN ACTIVE AST OR LEAVING ASTS DISABLED.
                     0028   178 ;
                     0028   179
                     0028   180 9LOCKED:                                    ;
                     0028   181
                     0028   182 ;
                     0028   183 ; THE AST DELIVERY INTERRUPT WAS SPURIOUS, A PENDING IPL-2 INTERRUPT LEFT
                     0028   184 ; OVER FROM THE PREVIOUS PROCESS.  THESE OCCUR INFREQUENTLY AND ARE DETECTED
                     0028   185 ; BY COMPARING THE ACCESS MODE OF THE FIRST AST IN THE QUEUE WITH THE CURRENT
                     0028   186 ; MODE OF THE INTERRUPTED PSL.  SPURIOUS IPL-2 INTERRUPTS ARE ALSO DETECTED
                     0028   187 ; BY THE REMQUE ABOVE FINDING AN EMPTY QUEUE.
                     0028   188 ;
                     0028   189
                     0028   190 SPURIOUS:                                   ;
10 A4   65   0E      0028   191          INSQUE  (R5),PCB$L_ASTQFL(R4)      ; REQUEUE AT HEAD OF QUEUE
        03   11      002C   192          BRB     SETLVL                     ; AND SET NEW ASTLEVEL
                     002E   193
                     002E   194 ;
                     002E   195 ; THE AST QUEUE IS NOW EMPTY.   EITHER THE AST DELIVERY INTERRUPT IS SPURIOUS
                     002E   196 ; OR ALL OF THE QUEUED ASTS HAVE BEEN CANCELED BY SIMPLY REMOVING THEM FROM THE
                     002E   197 ; QUEUE.  INSURE THAT ASTLVL IS SET TO PREVENT FURTHER INTERRUPTS.
                     002E   198 ;
                     002E   199 ; R4 - PCB ADDRESS
                     002E   200 ;
                     002E   201 MPH$QEMPTYCONT::                            ; MULTI-PROCESSING CHECK RETURNS TO HERE
                     002E   202 QEMPTY:
     51   04         D0     002E   203          MOVL    #4,R1              ; SET NULL AST LEVEL
                     0031   204 ;
                     0031   205 ; SET AST LEVEL TO BEST ESTIMATE OF CORRECT ACCESS MODE.
                     0031   206 ;
                     0031   207 ; R1 - NEW ACCESS MODE TO SET IN ASTLVL
                     0031   208 ;
  50   6C A4         D0     0031   209 SETLVL: MOVL    PCB$L_PHD(R4),R0   ; GET PROCESS HEADER ADDRESS
00CF C0   51         90     0035   210          MOVB    R1,PHD$B_ASTLVL(R0) ; SET ASTLEVEL IN HW PCB
     13   51         DA     003A   211          MTPR    R1,#PR$_ASTLVL     ; AND PROCESSOR REGISTER
                     003D   212 ASTDEXIT:                                   ; AST DELIVERY EXIT
     50   8E         7D     003D   213          MOVQ    (SP)+,R0           ; RESTORE REGISTERS R0,R1
     52   8E         7D     0040   214          MOVQ    (SP)+,R2           ; RESTORE REGISTERS R2,R3
     54   8E         7D     0043   215          MOVQ    (SP)+,R4           ; RESTORE REGISTERS R4,R5
                     02     0046   216          REI                        ; AND RETURN
```

```
                                0047   218          .SBTTL  NORM - NORMAL AST DELIVERY
                                0047   219 ;
                                0047   220 ;       AT THIS POINT THE KERNEL STACK IS:
                                0047   221 ;       00(SP) = SAVED R0
                                0047   222 ;       04(SP) = SAVED R1
                                0047   223 ;       08(SP) = SAVED R2
                                0047   224 ;       12(SP) = SAVED R3
                                0047   225 ;       16(SP) = SAVED R4
                                0047   226 ;       20(SP) = SAVED R5
                                0047   227 ;       24(SP) = SAVED PC
                                0047   228 ;       28(SP) = SAVED PSL
                                0047   229 ;
                                0047   230 ;       R0 - PHD ADDRESS
                                0047   231 ;       R1 - ACCESS MODE OF NEXT ACB OR NULL AST LEVEL
                                0047   232 ;       R4 - PCB ADDRESS
                                0047   233 ;       R5 - ACB ADDRESS
                                0047   234 ;
                                0047   235
                                0047   236 NORM:                                          ; NORMAL AST DELIVERY
                                0047   237          ASSUME  ACB$V_MODE EQ 0
                                0047   238          ASSUME  ACB$S_MODE EQ 2
                         51     D4     0047   239          CLRL    R1                      ; BACKGROUND R3 WITH ZEROES
      51    0B A5     FC 8F     8B     0049   240          BICB3   #^C<3>,ACB$B_RMOD(R5),R1; EXTRACT ACCESS MODE FOR CURRENT AST
   51    1C AE     02    18     ED     004F   241          CMPZV   #PSL$V_CURMOD,#PSL$S_CURMOD,28(SP),R1  ; IS CURRENT MODE LEGAL
                           D1     19     0055   242          BLSS    SPURIOUS                ; BR IF SPURIOUS
                      53   81     9E     0057   243          MOVAB   (R1)+,R3                ; SET FOR NEXT ACCESS MODE
            C9 0D A4  53     E1     005A   244          BBC     R3,PCB$B_ASTEN(R4),BLOCKED ; BR IF AST DISABLED
            C4 0C A4  53     E2     005F   245          BBSS    R3,PCB$B_ASTACT(R4),BLOCKED ; SET AST ACTIVE
                      13   51     DA     0064   246          MTPR    R1,#PR$_ASTLVL          ; SET AST LEVEL IN PROCESSOR REGISTER
                  50    6C A4     D0     0067   247          MOVL    PCB$L_PHD(R4),R0        ; GET PROCESS HEADER ADDRESS
               00CF C0    51     90     0068   248          MOVB    R1,PHD$B_ASTLVL(R0)     ; SET AST LEVEL IN HW PCB
            03 0B A5     06     E1     0070   249          BBC     #ACB$V_QUOTA,ACB$B_RMOD(R5),30$ ; SKIP IF NO QUOTA ACCOUNTING
                     38 A4     B6     0075   250          INCW    PCB$W_ASTCNT(R4)        ; UPDATE OUTSTANDING COUNT
                                0078   251 30$:                                          ; AND DELIVER AST
                                0078   252          SETIPL  #IPL$_ASTDEL                ; NOW DROP IPL TO PERMIT SYSTEM EVENTS
                                0078   253 ;
                                0078   254 ;       A NEW VALUE FOR ASTLVL HAS NOW BEEN COMPUTED AND SET.
                                0078   255 ;       THE AST REPRESENTED BY THE AST CONTROL BLOCK LOCATED VIA
                                0078   256 ;       R5 CAN NOW BE DELIVERED.
                                0078   257 ;
                           53   D5     0078   258          TSTL    R3                      ; CHECK FOR DELIVERY TO KERNEL
                           6A   12     007D   259          BNEQ    NOTKMODE                ; BR IF NOT KERNEL MODE
```

```
                                007F   261          .SBTTL  NORMAL KERNEL MODE AST
                                007F   262 ;
                                007F   263 ;       DELIVER NORMAL AST TO KERNEL MODE
                                007F   264 ;
                     50    8E   7D   007F   265 KMODE:  MOVQ    (SP)+,R0                ; RESTORE R0,R1
          000000D3'8F    10   AE   D1   0082   266        CMPL    16(SP),#ASTEXIT_CHMK    ; DELIVERY OCCUR DURING ASTEXIT REI?
                               06   12   008A   267        BNEQ    40$                     ; NO, JUST CONTINUE
                          17   AE   03   93   008C   268        BITB    #<PSL$M_CURMOD-         ; WAS INTERRUPTED MODE KERNEL?
                                                  0090   269                @<-PSL$V_CURMOD>>,23(SP);
                               0C   13   0090   270        BEQL    50$                     ; YES, PREVIOUS AST R0,R1,PC,PSL ALREADY ON
                                                  0092   271 ;
                                                  0092   272 ;       00(SP) = R2, 04(SP) = R3, 08(SP) = R4, 12(SP) = R5,
                                                  0092   273 ;       16(SP) = PC, 20(SP) = PSL
                                                  0092   274 ;
                                                  0092   275 40$:
                     7E    08   AE   7D   0092   276        MOVQ    8(SP),-(SP)             ; SHUFFLE STACK
                                                  0096   277 ;
                                                  0096   278 ;       00(SP) = R4, 04(SP) = R5, 08(SP) = R2, 12(SP) = R3,
                                                  0096   279 ;       16(SP) = R4, 20(SP) = R5, 24(SP) = PC, 28(SP) = PSL
                                                  0096   280 ;
                     7E    08   AE   7D   0096   281        MOVQ    8(SP),-(SP)             ; OPEN FOR AST ARG LIST
                                                  009A   282 ;
                                                  009A   283 ;       00(SP) = R2, 04(SP) = R3, 08(SP) = R4, 12(SP) = R5,
                                                  009A   284 ;       16(SP) = R2, 20(SP) = R3, 24(SP) = R4, 28(SP) = R5,
                                                  009A   285 ;       32(SP) = PC, 36(SP) = PSL
                                                  009A   286 ;
                     18   AE    50   7D   009A   287        MOVQ    R0,24(SP)               ; SET R0,R1 IN ARG LIST
                14   AE    14   A5   D0   009E   288 50$:    MOVL    ACB$L_ASTPRM(R5),20(SP) ; SET AST PARAMETER IN ARG LIST
                     10   AE    05   D0   00A3   289        MOVL    #5,16(SP)               ; SET COUNT FOR ARGUMENT LIST
                          50    55   D0   00A7   290        MOVL    R5,R0                   ; RELEASE AST CONTROL BLOCK
                            10   A5   DD   00AA   291        PUSHL   ACB$L_AST(R5)           ; SAVE AST ROUTINE ADDRESS
          05    08   A5    04   E1   00AD   292        BBC     #ACB$V_PKAST,ACB$B_RMOD(R5),60$ ; BR IF NO PIGGY-BACK KAST
                                                  00B2   293 ;
                                                  00B2   294 ; CALL PIGGY-BACK SPECIAL KERNEL AST ROUTINE.
                                                  00B2   295 ;       R5 - ACB ADDRESS (MUST BE PRESERVED)
                                                  00B2   296 ;       IPL = IPL$_ASTDEL (MUST NOT BE LOWERED)
                                                  00B2   297 ;
                            18   B5   16   00B2   298        JSB     @ACB$L_KAST(R5)         ; CALL KAST ROUTINE
                               08   11   00B5   299        BRB     70$                     ; NO DELETE FOR PKAST
          03    08   A5    05   E0   00B7   300 60$:    BBS     #ACB$V_NODELETE,ACB$B_RMOD(R5),70$; BR IF NOT DELETEABLE
                          FF41'   30   00BC   301        BSBW    EXE$DEANONPAGED         ; TO DYNAMIC POOL
                          51    8E   7D   00BF   302 70$:    MOVQ    (SP)+,R1                ; RESTORE R1,R2
                          53    8E   7D   00C2   303        MOVQ    (SP)+,R3                ; RESTORE R3,R4
                          55    8E   D0   00C5   304        MOVL    (SP)+,R5                ; RESTORE R5
                                           00C8   305        SETIPL  #0                      ; DROP IPL TO ZERO
                                           00CB   306 ;      BRB     EXE$ASTDEL              ; FALL THROUGH TO CALL AST ROUTINE
```

```
                     00CB   308 ;
                     00CB   309 ;       CALL AST ROUTINE WITH AST ARGUMENT LIST
                     00CB   310 ;
                     00CB   311 ;       THE CALL IS EXECUTED AT THE MODE WHICH RECEIVED THE AST WITH
                     00CB   312 ;       THE AST ARGUMENT LIST ON THE TOP OF THE STACK.  WHEN THE
                     00CB   313 ;       AST ROUTINE RETURNS FROM THE CALL, AN ASTEXIT CHANGE MODE
                     00CB   314 ;       TO KERNEL INSTRUCTION WILL BE ISSUED.  ASTEXIT WILL RESET
                     00CB   315 ;       THE AST ACTIVE BIT FOR THE CURRENT MODE AND MAY CAUSE DELIVERY
                     00CB   316 ;       OF ADDITIONAL ASTS.
                     00CB   317 ;
                     00CB   318 ;       AST ARGUMENT LIST:
                     00CB   319 ;       ------------------
                     00CB   320 ;
                     00CB   321 ;       00(SP) = NUMBER OF ARGUMENTS, =5
                     00CB   322 ;       04(SP) = AST PARAMETER
                     00CB   323 ;       08(SP) = SAVED R0
                     00CB   324 ;       12(SP) = SAVED R1
                     00CB   325 ;       16(SP) = SAVED PC
                     00CB   326 ;       20(SP) = SAVED PSL
                     00CB   327 ;
                     00CB   328 EXE$ASTDEL::                         ; DELIVER AST CALL
 61    6E    FA      00CB   329         CALLG   (SP),(R1)            ; CALL AST ROUTINE
                     00CE   330 EXE$ASTRET::                         ; RETURN ADDRESS FOR AST CALL
 5E    08    C0      00CE   331         ADDL    #8,SP                ; REMOVE ARG COUNT AND ASTPRM
       00    BC      00D1   332         CHMK    S^#ASTEXIT           ; AND EXIT FROM AST ROUTINE
                     00D3   333 ASTEXIT_CHMK:                        ; RETURN ADDRESS FOR AST EXIT CHMK
 50    8E    7D      00D3   334         MOVQ    (SP)+,R0             ; RESTORE R0,R1
             02      00D6   335         REI                          ; EXECUTE REI IN MODE OF AST
                     00D7   336
```

```
                                00D7   338          .SBTTL  NORMAL EXEC, SUPER AND USER MODE AST
                                00D7   339
                                00D7   340          .ENABLE LSB
                                00D7   341 ;
                                00D7   342 ; SPECIAL CASE CODE TO HANDLE THE SITUATION WHERE THE OUTER MODE SP HAS
                                00D7   343 ; BEEN FOUND TO BE BELOW THE FP, MEANING THE STACK IS CORRUPTED. THIS
                                00D7   344 ; SITUATION CAN RESULT FROM A 780 MICROCODE BUG SHICH LEAVES THE SP
                                00D7   345 ; WRONG WHEN A RET PAGEFAULTS WHILE LOOKING AT THE ARG LIST. WE CHECK
                                00D7   346 ; FOR THIS CASE AND SET SP = FP, WHICH IS ADEQUATE TO SERVICE THE AST
                                00D7   347 ; AND LET THE RET RE-EXECUTE.
                                00D7   348 ;
     50    5D    00000048 8F    C1  00D7   349 1$:     ADDL3   #18*4,FP,R0             ; RANGE LIMIT IS MAXIMAL FRAME + ALIGNMENT
                       50    51    D1  00DF   350         CMPL    R1,R0               ; CHECK IF SP IS WITHIN RANGE
                             0D    1A  00E2   351         BGTRU   5$                  ; IF NOT, SOME OTHER PROBLEM - DON'T FIX
                       51    5D    D0  00E4   352         MOVL    FP,R1               ; SET OUTER MODE SP = FP
                             08    11  00E7   353         BRB     5$                  ; AND CONTINUE
                                       00E9   354
                                       00E9   355 ;
                                       00E9   356 ;       DELIVER NORMAL AST FOR EXEC, SUPER AND USER MODE
                                       00E9   357 ;
                                       00E9   358
                                       00E9   359 NOTKMODE:                           ; NOT AN AST FOR KERNEL MODE
                                       00E9   360 ;
                       51    53    DB  00E9   361         MFPR    R3,R1               ; GET STACK POINTER
                       5D    51    D1  00EC   362         CMPL    R1,FP               ; SEE IF STACK POINTER IS ABOVE FP
                             E6    1A  00EF   363         BGTRU   1$                  ; BRANCH IF NOT - POSSIBLE FIXUP
                                       00F1   364 5$:     IFNOWRT #24,-24(R1),STACKERR,R3 ; ENOUGH STACK SPACE??
                 71    18    AE    7D  00F8   365         MOVQ    24(SP),-(R1)        ; MOVE PC,PSL TO PROPER STACK
                       71    8E    7D  00FC   366         MOVQ    (SP)+,-(R1)         ; AND R0,R1 FROM KERNEL STACK
000000D3'8F    10    AE    D1  00FF   367         CMPL    16(SP),#ASTEXIT_CHMK        ; DELIVERY OCCUR DURING ASTEXIT REI?
                             3B    13  0107   368         BEQL    5$                  ; YES, CHECK FOR SAME MODE
                 71    14    A5  D0  0109   369 10$:     MOVL    ACB$L_ASTPRM(R5),-(R1) ; SET AST PARAMETER IN ARG LIST
                       71    05  D0  010D   370         MOVL    #5,-(R1)            ; AND FINALLY, ARGUMENT COUNT OF 5
                       53    51  DA  0110   371         MTPR    R1,#PR$_KSP         ; SAVE UPDATED STACK POINTER
                       10    A5  DD  0113   372         PUSHL   ACB$L_AST(R5)       ; STACK AST ENTRY POINT
                       50    55  D0  0116   373         MOVL    R5,R0               ; SET ADDRESS OF ACB FOR RELEASE
           14 AE       AF    AF  9E  0119   374         MOVAB   EXE$ASTDEL,20(SP)   ; SET PC TO AST DELIVERY CALL
                 53    6343      DE  011E   375         MOVAL   (R3)[R3],R3         ; MODE=MODE*5, CURMOD=PRVMOD
                                   0122   376         ASSUME  PSL$V_CURMOD EQ PSL$V_PRVMOD+2; FOR ABOVE MOVAL
     18 AE       53    16    78  0122   377         ASHL    #PSL$V_PRVMOD,R3,24(SP) ; SYNTHESIZE PSL FOR PROPER MODE
              0B   A5    30    93  0127   378         BITB    #<ACB$M_NODELETE!ACB$M_PKAST>,ACB$B_RMOD(R5) ; SPECIAL ACTIONS?
                          0D    12  012B   379         BNEQ    40$                 ; BR IF SO AND DECODE
                    FED0'       30  012D   380 20$:     BSBW    EXE$DEANONPAGED     ; RELEASE AST CONTROL BLOCK
                       51    8E    7D  0130   381 30$:     MOVQ    (SP)+,R1            ; RESTORE R1,R2
                       53    8E    7D  0133   382         MOVQ    (SP)+,R3            ; RESTORE R3,R4
                       55    8E    D0  0136   383         MOVL    (SP)+,R5            ; RESTORE R5
                             02    11  0139   384         REI                         ; AND ENTER AST MODE
                                       013A   385                                     ; DROPS IPL TO ZERO
        F1 0B A5          04    E1  013A   386 40$:     BBC     #ACB$V_PKAST,ACB$B_RMOD(R5),30$ ; BR IF NO PIGGY-BACK KAST
                                   013F   387 ;
                                   013F   388 ; CALL PIGGY-BACK SPECIAL KERNEL AST ROUTINE.
                                   013F   389 ;       R5 - ACB ADDRESS (MUST BE PRESERVED)
                                   013F   390 ;       IPL = IPL$_ASTDEL (MUST NOT BE LOWERED)
                                   013F   391 ;
                    18 B5    16    013F   392         JSB     @ACB$L_KAST(R5)     ; CALL KAST ROUTINE
                       EC    11  0142   393         BRB     30$                 ; NO DELETE FOR PKAST
                                   0144   394 ;
```

```
                                0144   395 ; SPECIAL CASE FOR AST DURING AST EXIT.
                                0144   396 ;
                     18   ED   0144   397 50$:    CMPZV   #PSL$V_CURMOD,-          ; WAS AST'S MODE THE ONE INTERRUPTED?
53   14 AE              02      0146   398                 #PSL$S_CURMOD,20(SP),R3 ;
                        BD   12 0148   399         BNEQ    10$                      ; NO, JUST CONTINUE
                     51 53   DB 014A   400         MFPR    R3,R1                    ; RESET SP - R0,R1,PC,PSL ALREADY ON STACK
                        B8   11 014D   401         BRB     10$                      ; CONTINUE
                                014F   402
                                014F   403         .DISABLE LSB
                                014F   404
                                014F   405 ;
                                014F   406 ;       REFLECT STACK ERROR
                                014F   407 ;
                                014F   408 STACKERR:                                ; ERROR IN STACK MOVE
                     53   03   D1 014F   409         CMPL    #PSL$C_USER,R3           ; IS THIS AST FOR USER MODE?
                        11   12 0152   410         BNEQ    10$                      ; NO, THEN WE CANT EXTEND THE STACK
                        3E   BB 0154   411         PUSHR   #^M<R1,R2,R3,R4,R5>      ; SAVE NECESSARY REGISTERS
                  52    E8   A1 9E 0156   412         MOVAB   -24(R1),R2               ; COMPUTE DESIRED STACK TOP ADDRESS
               00000000'EF   16 015A   413         JSB     EXE$EXPANDSTK            ; EXPAND USER STACK TO DESIRED SIZE
                        3E   BA 0160   414         POPR    #^M<R1,R2,R3,R4,R5>      ; RESTORE REGISTERS
                     82 50   E8 0162   415         BLBS    R0,NOTKMODE              ; CONTINUE IF SPACE CREATED
                00 0C A4     53   E5 0165   416 10$:     BBCC    R3,PCB$B_ASTACT(R4),20$ ; CLEAR AST ACTIVE BIT
                   7E    10 AE   7D 016A   417 20$:     MOVQ    16(SP),-(SP)             ; CREATE SPACE ON STACK
                   7E    10 AE   7D 016E   418         MOVQ    16(SP),-(SP)             ; BY MOVING R2-R5 DOWN
                20 AE    28 AE   7D 0172   419         MOVQ    40(SP),32(SP)            ; SAVE PC,PSL AT INTERRUPT
                28 AE    10 A5   D0 0177   420         MOVL    ACB$L_AST(R5),40(SP)     ; SET PC AT FAULT TO AST ADDRESS
                1C AE    14 A5   D0 017C   421         MOVL    ACB$L_ASTPRM(R5),28(SP)  ; SET ASTPRM IN ARGLIST
                   18 AE    51   D0 0181   422         MOVL    R1,24(SP)                ; SAVE STACK VA AT FAULT
                      53    05   C4 0185   423         MULL    #1+<1@<PSL$V_CURMOD-PSL$V_PRVMOD>>,R3   ; CURRENT MODE = PREV
             2C AE    53    16   9C 0188   424         ROTL    #PSL$V_PRVMOD,R3,44(SP)  ; SYNTHESIZE NEW PSL FOR FAULT
                      50    55   D0 018D   425         MOVL    R5,R0                    ; SET ADDRESS FOR RELEASE OF ACB
                05 0B A5    04   E1 0190   426         BBC     #ACB$V_PKAST,ACB$B_RMOD(R5),30$ ; BR IF NO PIGGY-BACK KAST
                                0195   427 ;
                                0195   428 ; CALL PIGGY-BACK SPECIAL KERNEL AST ROUTINE.
                                0195   429 ;       R5 - ACB ADDRESS (MUST BE PRESERVED)
                                0195   430 ;       IPL = IPL$_ASTDEL (MUST NOT BE LOWERED)
                                0195   431 ;
                      18 B5     16   0195   432         JSB     @ACB$L_KAST(R5)          ; CALL KAST ROUTINE
                         08     11   0198   433         BRB     40$                      ; NO DELETE FOR PKAST
                03 0B A5    05     E0 019A   434 30$:     BBS     #ACB$V_NODELETE,ACB$B_RMOD(R5),40$; BR IF NOT DELETEABLE
                      FE5C'    30   019F   435         BSBW    EXE$DEANONPAGED          ; AND DEALLOCATE IT
                         3C     BA 01A2   436 40$:     POPR    #^M<R2,R3,R4,R5>         ; RESTORE ALL REGISTERS
                         03     BA 01A4   437         POPR    #^M<R0,R1>               ; FROM POINT OF INTERRUPT
                                01A6   438         SETIPL  #0                       ; DROP IPL TO 0
                      FE52'    31   01A9   439         BRW     EXE$ASTFLT               ; REFLECT EXCEPTION
```

```
                     01AE   441          .SBTTL  EXE$IPAPBKAST - SPECIAL PIGGY BACK KAST FOR IPAST SERVICE
                     01AE   442 ;++
                     01AE   443 ; FUNCTIONAL DESCRIPTION:
                     01AE   444 ;       THIS ROUTINE IS A SPECIAL PIGGY BACK KAST ROUTINE CALLED BY
                     01AE   445 ;       ASTDEL DURING DELIVERY OF THE INTER-PROCESS AST.  IT TAKES THE
                     01AE   446 ;       IPAST INDEX RESIDING IN ACB$L_AST AND USES IT TO INDEX INTO THE
                     01AE   447 ;       VECTOR OF AST ADDRESSES LOCATED IN THE VECTOR PAGE OF THE CONTROL
                     01AE   448 ;       REGION.  THIS ADDRESS REPLACES WHAT WAS ON THE STACK WHEN THIS
                     01AE   449 ;       ROUTINE WAS ENTERED.  ROUTINE IS INCLUDED HERE SINCE ITS HAS
                     01AE   450 ;       KNOWLEDGE OF HOW STACK LOOKS WHEN PIGGYBACK AST IS CALLED.
                     01AE   451 ;
                     01AE   452 ; INPUTS:
                     01AE   453 ;       4(SP) --> AST ADDRESS
                     01AE   454 ;       ACB$L_AST(R5) = INDEX OF IPASTS
                     01AE   455 ;
                     01AE   456 ; OUTPUTS:
                     01AE   457 ;       4(SP) --> NEW AST ADDRESS
                     01AE   458 ;
                     01AE   459 ;--
                     01AE   460
                     01AE   461 EXF$IPAPBKAST::
          52   10 A5   D0  01AE   462          MOVL    ACB$L_AST(R5),R2        ; Pick up index
50   00000000'9F42   D0  01B2   463          MOVL    @#CTL$AL_IPASTVEC[R2],R0; Get IPAST address
            04   13  01BA   464          BEQL    10$                     ; No longer in use
      04 AE   50   D0  01BC   465          MOVL    R0,4(SP)                ; Fixup AST delivery address
         50   55   D0  01C0   466 10$:     MOVL    R5,R0                   ; Transfer address of packet
             FE3A'  31  01C3   467          BRW     EXE$DEANONPAGED         ; Get rid of it
```

```
                        01C6   469          .SBTTL  SCH$QAST - ENQUEUE AST CONTROL BLOCK FOR PROCESS
                        01C6   470 ;++
                        01C6   471 ; FUNCTIONAL DESCRIPTION:
                        01C6   472 ;       SCH$QAST INSERTS THE AST C NTROL BLOCK SUPPLIED IN THE PROPER
                        01C6   473 ;       POSITION BY ACCESS MODE IN THE AST QUEUE OF THE PROCESS SPECIFIED
                        01C6   474 ;       BY THE PID FIELD OF THE AST CONTROL BLOCK.  AN AST ARRIVAL EVENT
                        01C6   475 ;       IS THEN REPORTED FOR THE PROCESS TO REACTIVATE FROM A WAIT STATE
                        01C6   476 ;       IF APPROPRIATE.  THE AST CONTROL BLOCK WILL BE RELEASED IMMEDIATELY
                        01C6   477 ;       IF THE PID SPECIFIES A NON-EXISTENT PROCESS.
                        01C6   478 ;
                        01C6   479 ;       LOADABLE MULTI-PROCESSING CODE WILL REPLACE THIS ROUTINE WITH
                        01C6   480 ;       ENTIRELY NEW CODE, AT MPH$QAST.
                        01C6   481 ;
                        01C6   482 ; CALLING SEQUENCE:
                        01C6   483 ;       BSB/JSB SCH$QAST
                        01C6   484 ;
                        01C6   485 ; INPUT PARAMETERS:
                        01C6   486 ;       R2 - PRIORITY INCREMENT CLASS
                        01C6   487 ;       R5 - POINTER TO AST CONTROL BLOCK
                        01C6   488 ;
                        01C6   489 ; IMPLICIT INPUTS:
                        01C6   490 ;       PCB OF PROCESS IDENTIFIED BY PID FIELD
                        01C6   491 ;
                        01C6   492 ; OUTPUT PARAMETERS:
                        01C6   493 ;       R0 - COMPLETION STATUS CODE
                        01C6   494 ;       R4 - PCB ADDRESS OF PROCESS FOR WHICH AST WAS QUEUED
                        01C6   495 ;
                        01C6   496 ; SIDE EFFECTS:
                        01C6   497 ;       THE PROCESS IDENTIFIED BY THE PID IN THE AST CONTROL BLOCK
                        01C6   498 ;       WILL BE MADE EXECUTABLE IF NOT SUSPENDED.
                        01C6   499 ;
                        01C6   500 ; COMPLETION CODES:
                        01C6   501 ;       SS$_NORMAL  -  NORMAL SUCCESSFUL COMPLETION STATUS
                        01C6   502 ;       SS$_NONEXPR -  NON-EXISTENT PROCESS
                        01C6   503 ;--
                        01C6   504          .ENABL  LSB
                        01C6   505 QNONEXPR:
                        01C6   506          ; DEALLOCATE THE ACB AS LONG AS THE NODELETE BIT IS NOT SET.
                        01C6   507          ; THIS REALLY SHOULDN'T HAPPEN, BUT IF IT DOES, WE CHOOSE
                        01C6   508          ; TO POSSIBLY LOSE POOL OVER POSSIBLY CORRUPTING IT.
                        01C6   509
06 0B A5    05    E0    01C6   510          BBS     #ACB$V_NODELETE,ACB$B_RMOD(R5),5$; BR IF NOT DELETEABLE
      50    55    D0    01CB   511          MOVL    R5,R0                       ; RELEASE AST CONTROL BLOCK
          FE2F'   30    01CE   512          BSBW    EXE$DEANONPAGED             ; IF NO SUCH PROCESS
 50   08E8 8F    3C    01D1   513 5$:       MOVZWL  #SS$_NONEXPR,R0             ; SET ERROR STATUS CODE
            43    11    01D6   514          BRB     QEXIT                       ; AND EXIT
                        01D8   515
                        01D8   516 MPH$QAST::                                  ; MULTI-PROCESSING HOOK TO REPLACE RTN
                        01D8   517 SCH$QAST::                                  ; ENQUEUE AST FOR PROCESS
   50   0C A5    3C    01D8   518          MOVZWL  ACB$L_PID(R5),R0            ; GET PROCESS INDEX FOR AST TARGET
                        01DC   519          DSBINT  #IPL$_SYNCH                 ; DISABLE SYSTEM EVENTS
54   0000'DF40    D0    01E2   520          MOVL    @W^SCH$GL_PCBVEC[R0],R4     ; LOOK UP PCB ADDRESS
60 A4   0C A5    D1    01E8   521          CMPL    ACB$L_PID(R5),PCB$L_PID(R4) ; CHECK FOR MATCH IN PID
            D7    12    01ED   522          BNEQ    QNONEXPR                    ; PID MISMATCHES
            50    D4    01EF   523          CLRL    R0                          ; ASSUME KERNEL MODE AND CLEAR HIGH BITS
   10 A4    65    0E    01F1   524          INSQUE  (R5),PCB$L_ASTQFL(R4)       ; ASSUME QUEUE IS EMPTY AND ATTEMPT INSERT
            2D    12    01F5   525          BNEQ    50$                         ; BR IF IT WAS NOT EMPTY
```

```
                0B A5   95   01F7   526          TSTB    ACB$B_RMOD(R5)          ; CHECK FOR SPECIAL KERNEL AST
                   06   19   01FA   527          BLSS    10$                     ; BR IF YES
      50  0B A5   FC 8F   8B   01FC   528          BICB3   #^C<3>,ACB$B_RMOD(R5),R0; GET AST MODE
                                0202   529 ;
                                0202   530 ; THE PROCESS HEADER ADDRESS IS ALWAYS A SYSTEM SPACE ADDRESS (NEGATIVE NUMBER)
                                0202   531 ; WHILE THE PROCESS HEADER IS RESIDENT.  DURING THE OUTSWAP TRANSITION IT IS
                                0202   532 ; THE BALANCE SLOT INDEX, A SMALL POSITIVE NUMBER.  FINALLY, AFTER OUTSWAP IT
                                0202   533 ; IS SET TO ZERO.  HENCE, THE FOLLOWING TEST COMBINES THE FETCH OF THE PHD
                                0202   534 ; ADDRESS WITH THE TEST FOR PROCESS RESIDENCE.
                                0202   535 ;
        51   6C A4   D0   0202   536 10$:    MOVL    PCB$L_PHD(R4),R1        ; POINT TO PROCESS HEADER
                   0C   18   0206   537          BGEQ    25$                     ; DON'T SET ASTLVL IF NOT RESIDENT
    00CF C1    50   90   0208   538          MOVB    R0,PHD$B_ASTLVL(R1)     ; SET ASTLVL IN PROCESS HEADER
    54   0000'CF   D1   020D   539          CMPL    W^SCH$GL_CURPCB,R4      ; IS PROCESS CURRENT PROCESS
                   0B   13   0212   540          BEQL    40$                     ; YES,
                                0214   541 25$:    RPTEVT  AST                     ; REPORT AST ARRIVAL
        50   01   3C   0218   542 30$:    MOVZWL  #SS$_NORMAL,R0          ; SET SUCCESS STATUS CODE
                                021B   543 QEXIT:  ENBINT                       ; ENABLE INTERRUPTS
                   05   021E   544          RSB                             ; AND RETURN
                                021F   545 ;
                                021F   546 ; IF THE AST IS BEING ENQUEUED FOR THE CURRENT PROCESS, THEN THE REPORTING
                                021F   547 ; OF THE AST EVENT CAN BE BYPASSED AND THE ASTLVL PROCESSOR REGISTER MUST BE
                                021F   548 ; SET INSTEAD.
                                021F   549 ;
        13   50   DA   021F   550 40$:    MTPR    R0,#PR$_ASTLVL          ; ALSO SET ASTLVL REGISTER
                   F4   11   0222   551          BRB     30$                     ;
                                0224   552 ;
                                0224   553 ; THE AST QUEUE WAS NOT EMPTY (ITS USUAL CONDITION) AND THE PROPER
                                0224   554 ; POSITION FOR THE NEW AST MUST BE LOCATED.  SINCE THE AST CONTROL
                                0224   555 ; BLOCK HAS BEEN ERRONEOUSLY INSERTED ON THE QUEUE, IT MUST BE REMOVED
                                0224   556 ; FIRST.
                                0224   557 ;
        55   65   0F   0224   558 50$:    REMQUE  (R5),R5                 ; ELSE CORRECT MISTAKE
    51   10 A4   DE   0227   559          MOVAL   PCB$L_ASTQFL(R4),R1     ; POINT TO QUEUE HEADER
        53   61   D0   022B   560          MOVL    (R1),R3                 ; GET FIRST ENTRY ON QUEUE
          0B A5   95   022E   561          TSTB    ACB$B_RMOD(R5)          ; CHECK FOR SPECIAL KERNEL AST
                   0F   18   0231   562          BGEQ    70$                     ; BR IF NOT
                                0233   563 ;
                                0233   564 ; THE NEW AST IS A SPECIAL KERNEL AST.  IT WILL GO AFTER ALL OTHER SPECIAL
                                0233   565 ; KERNEL ASTS OR AT THE HEAD OF THE QUEUE IF THERE ARE NONE.
                                0233   566 ;
        53   51   D1   0233   567 60$:    CMPL    R1,R3                   ; CHECK FOR END OF QUEUE
                   27   13   0236   568          BEQL    110$                    ; BR IF NOT
          0B A3   95   0238   569          TSTB    ACB$B_RMOD(R3)          ; CHECK FOR SPECIAL KERNEL IN QUEUE
                   22   18   023B   570          BGEQ    110$                    ; BR IF NOT
        53   63   D0   023D   571          MOVL    (R3),R3                 ; FLINK ON TO NEXT ACB
                   F1   11   0240   572          BRB     60$                     ;
                                0242   573 ;
                                0242   574 ; THE NEW AST IS A NORMAL AST.  IT WILL GO AFTER ALL SPECIAL KERNEL ASTS
                                0242   575 ; AND ASTS WITH LOWER ACCESS MODE.
                                0242   576 ;
      50  0B A5   FC 8F   8B   0242   577 70$:    BICB3   #^C<3>,ACB$B_RMOD(R5),R0; GET AST MODE
        53   51   D1   0248   578 80$:    CMPL    R1,R3                   ; CHECK FOR END OF QUEUE
                   12   13   024B   579          BEQL    110$                    ; INSERT IF AT END
        02   00   ED   024D   580          CMPZV   #ACB$V_MODE,#ACB$S_MODE,-
      50   0B A3        0250   581                  ACB$B_RMOD(R3),R0       ; COMPARE ACCESS MODES
                   05   14   0253   582          BGTR    100$                    ; IF GTR AT RIGHT PLACE
```

```
                53   63   D0  0255   583 90$:     MOVL    (R3),R3                 ; FLINK ON TO NEXT ACB
                     EE   11  0258   584          BRB     80$                     ;
                  0B A3   95  025A   585 100$:    TSTB    ACB$B_RMOD(R3)          ; IS THIS ENTRY A SPECIAL KAST?
                     F6   19  025D   586          BLSS    90$                     ; YES, MUST GO AFTER THIS
                          025F   587 ;
                          025F   588 ; NOW THE CORRECT POSITION HAS BEEN LOCATED.  INSERT THE AST CONTROL BLOCK
                          025F   589 ; ON THE QUEUE AND COMPUTE THE NEW VALUE FOR ASTLVL BY INTERROGATING THE
                          025F   590 ; MODE OF THE AST CONTROL BLOCK AT THE HEAD OF THE QUEUE.
                          025F   591 ;
             04 B3    65   0E  025F   592 110$:    INSQUE  (R5),@ACB$L_ASTQBL(R3)  ; INSERT AFTER PREVIOUS
                      50   D4  0263   593          CLRL    R0                      ; ASSUME KERNEL MODE
             51    10 A4   D0  0265   594          MOVL    PCB$L_ASTQFL(R4),R1     ; GET HEAD OF AST QUEUE
                   0B A1   95  0269   595          TSTB    ACB$B_RMOD(R1)          ; IS IT KAST?
                      94   19  026C   596          BLSS    10$                     ; BR IF YES TO SET ASTLVL
     50    0B A1    FC 8F   8B  026E   597          BICB3   #^C<3>,ACB$B_RMOD(R1),R0 ; GET AST MODE FOR HEAD OF QUEUE
                      8C   11  0274   598          BRB     10$                     ; GO SET ASTLVL
                           0276   599
                           0276   600          .DSABL LSB
                           0276   601          ASSUME  ACB$V_MODE EQ 0
                           0276   602          ASSUME  ACB$S_MODE EQ 2
                           0276   603          ASSUME  ACB$V_KAST EQ 7
                           0276   604
```

```
                                0276   606          .SBTTL  SCH$NEWLVL - COMPUTE NEW AST LEVEL
                                0276   607 ;
                                0276   608 ;       SCH$NEWLVL - COMPUTE NEW ASTLVL
                                0276   609 ;
                                0276   610 ;       THIS ROUTINE COMPUTES THE NEW AST LEVEL.  MPH$NEWLVLHK IS
                                0276   611 ;       THE LOCATION OF A HOOK FOR LOADABLE MULTI-PROCESSING CODE.
                                0276   612 ;       THE REMAINDER OF THE ROUTINE IS REPLACED BY THE LOADABLE CODE.
                                0276   613 ;
                                0276   614 ; INPUT:
                                0276   615 ;       R4 - ADDRESS OF CURRENT PCB
                                0276   616 ;
                                0276   617 ; OUTPUT:
                                0276   618 ;       PR$_ASTLVL - NEW AST LEVEL
                                0276   619 ;       PHD$V_ASTLVL - NEW AST LEVEL
                                0276   620 ;
                                0276   621          .ENABL  LSB
                                0276   622 SCH$NEWLVL::                                ; COMPUTE NEW AST LEVEL
          50    10 A4    DE     0276   623          MOVAL   PCB$L_ASTQFL(R4),R0        ; GET ADDRESS OF AST QUEUE
                                027A   624          DSBINT  #IPL$_SYNCH                ; DISABLE SYSTEM EVENTS
                52    D4     0280   625          CLRL    R2                         ; ASSUME KERNEL MODE
             51    60    D0     0282   626          MOVL    (R0),R1                    ; GET FLINK
             51    50    D1     0285   627          CMPL    R0,R1                      ; AND CHECK FOR EMPTY QUEUE
                   1B    13     0288   628          BEQL    20$                        ; YES, QUEUE IS EMPTY
                                028A   629          ASSUME  ACB$V_KAST EQ 7
          0B A1    95     028A   630          TSTB    ACB$B_RMOD(R1)             ; CHECK FOR KERNEL AST
                   06    19     028D   631          BLSS    10$                        ; BR IF NOT
    52    0B A1    FC 8F  8B     028F   632          BICB3   #^C<3>,ACB$B_RMOD(R1),R2; GET REQUEST MODE
          50    6C A4    D0     0295   633 10$:      MOVL    PCB$L_PHD(R4),R0           ; GET ADDRESS OF PHD
                                0299   634 MPH$NEWLVLHK::                             ; MULTI-PROCESSING HOOK REPLACES REST
                                0299   635                                            ; OF THIS ROUTINE
                                0299   636          ASSUME  PHD$S_ASTLVL EQ 8          ; FOR USE OF MOVB
  00CF  C0    52    90     0299   637          MOVB    R2,PHD$B_ASTLVL(R0)        ; SET ASTLVL IN PHD
        13    52    DA     029E   638          MTPR    R2,#PR$_ASTLVL             ; SET ASTLVL REGISTER
                            02A1   639          ENBINT                             ; ENABLE SYSTEM EVENTS AGAIN
                      05    02A4   640          RSB                                ; RETURN
                            02A5   641 ;
              52    04    D0     02A5   642 20$:      MOVL    #4,R2                      ; SET NULL AST LEVEL
                    EB    11     02A8   643          BRB     10$                        ;
                            02AA   644          .DSABL  LSB
```

```
                        02AA    646          .SBTTL  SCH$SWAPACBS - SWAP AST CONTROL BLOCKS
                        02AA    647          .SBTTL  SCH$REMOVACB - REMOVE AST CONTROL BLOCK
                        02AA    648
                        02AA    649 ;++
                        02AA    650 ;
                        02AA    651 ; FUNTIONAL DESCRIPTION:
                        02AA    652 ;
                        02AA    653 ;       SCH$SWAPACBS REMOVES AN ACB FROM AN AST QUEUE AND INSERTS ANOTHER
                        02AA    654 ;       ACB IN ITS PLACE.  I.E. IT SWAPS THE TWO ACBS.  THIS IS NORMALLY
                        02AA    655 ;       DONE BECAUSE THE ACB BEING REMOVED IS PART OF A LARGER STRUCTURE
                        02AA    656 ;       THAT MUST BE FREED UP FOR ANOTHER PURPOSE.  THE ACB BEING
                        02AA    657 ;       INSERTED IS NORMALLY A COPY OF THE ONE BEING REMOVED EXCEPT THAT THE
                        02AA    658 ;       ACB$M_NODELETE BIT IS NORMALLY CLEARED.  I.E. THE ONE BEING
                        02AA    659 ;       REMOVED WAS NOT DELETABLE BUT THE ONE BEING INSERTED IS.
                        02AA    660 ;
                        02AA    661 ;
                        02AA    662 ;       SCH$REMOVACB IS AN ALTERNATE ENTRY POINT THAT SIMPLY REMOVES
                        02AA    663 ;       AN ACB FROM AN AST QUEUE.
                        02AA    664 ;
                        02AA    665 ; CALLING SEQUENCE:
                        02AA    666 ;
                        02AA    667 ;       BSB/JSB SCH$SWAPACBS
                        02AA    668 ;       BSB/JSB SCH$REMOVACB
                        02AA    669 ;       NOTE:  THESE ROUTINES MUST BE CALLED AT IPL$_SYNCH
                        02AA    670 ;
                        02AA    671 ; INPUT PARAMETERS:
                        02AA    672 ;
                        02AA    673 ;       R2      ADDRESS OF ACB TO INSERT (SCH$SWAPACBS ONLY)
                        02AA    674 ;       R5      ADDRESS OF ACB TO REMOVE
                        02AA    675 ;
                        02AA    676 ; OUTPUT PARAMETERS:
                        02AA    677 ;
                        02AA    678 ;       NONE
                        02AA    679 ;--
                        02AA    680
                        02AA    681 SCH$SWAPACBS::
65    62    0E          02AA    682          INSQUE  (R2),(R5)              ; INSERT NEW ACB AFTER OLD ONE
                        02AD    683 SCH$REMOVACB::
55    65    0F          02AD    684          REMQUE  (R5),R5                ; REMOVE OLD ACB
                  05    02B0    685          RSB
                        02B1    686
                        02B1    687
                        02B1    688
                        02B1    689          .END
```

```
ACB$B_RMOD                = 0000000B
ACB$L_AST                 = 00000010
ACB$L_ASTPRM              = 00000014
ACB$L_ASTQBL              = 00000004
ACB$L_KAST                = 00000018
ACB$L_PID                 = 0000000C
ACB$M_NODELETE            = 00000020
ACB$M_PKAST               = 00000010
ACB$S_MODE                = 00000002
ACB$V_KAST                = 00000007
ACB$V_MODE                = 00000000
ACB$V_NODELETE            = 00000005
ACB$V_PKAST               = 00000004
ACB$V_QUOTA               = 00000006
ASTDEXIT                    0000003D R      02
ASTEXIT                   = 00000000
ASTEXIT_CHMK                000000D3 R      02
BLOCKED                     00000028 R      02
CTL$AL_IPASTVEC             ******** X      02
EVT$_AST                    ******** X      02
EXE$ASTDEL                  000000CB RG     02
EXE$ASTFLT                  ******** X      02
EXE$ASTRET                  000000CE RG     02
EXE$DEANONPAGED             ******** X      02
EXE$EXPANDSTK               ******** X      02
EXE$IPAPBKAST               000001AE RG     02
GETNEXT                     0000000C R      02
IPL$_ASTDEL               = 00000002
IPL$_SYNCH                = 00000008
KMODE                       0000007F R      02
MPH$ASTDELCONT              0000001A RG     02
MPH$ASTDELHK                00000014 R      02
MPH$NEWLVLHK                00000299 RG     02
MPH$QAST                    000001D8 RG     02
MPH$QEMPTYCONT              0000002E RG     02
NORM                        00000047 R      02
NOTKMODE                    000000E9 R      02
PCB$B_ASTACT              = 0000000C
PCB$B_ASTEN               = 0000000D
PCB$L_ASTQFL              = 00000010
PCB$L_PHD                 = 0000006C
PCB$L_PID                 = 00000060
PCB$W_ASTCNT              = 00000038
PHD$B_ASTLVL              = 000000CF
PHD$S_ASTLVL              = 00000008
PR$_ASTLVL                = 00000013
PR$_IPL                   = 00000012
PSL$C_USER                = 00000003
PSL$M_CURMOD              = 03000000
PSL$S_CURMOD              = 00000002
PSL$V_CURMOD              = 00000018
PSL$V_PRVMOD              = 00000016
QEMPTY                      0000002E R      02
QEXIT                       0000021B R      02
QNONEXPR                    000001C6 R      02
SCH$ASTDEL                  00000000 RG     02
SCH$GL_CURPCB               ******** X      02
SCH$GL_PCBVEC               ******** X      02
SCH$NEWLVL                  00000276 RG     02
SCH$QAST                    000001D8 RG     02
SCH$REMOVACB                000002AD RG     02
SCH$RSE                     ******** X      02
SCH$SWAPACBS                000002AA RG     02
SETLVL                      00000031 R      02
SPURIOUS                    00000028 R      02
SS$_NONEXPR               = 000008E8
SS$_NORMAL                = 00000001
STACKERR                    00000151 R      02
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
A$EXENONPAGED	000002B1 (689.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	39	00:00:00.09	00:00:01.02
Command processing	169	00:00:00.57	00:00:03.53
Pass 1	295	00:00:08.99	00:00:32.80
Symbol table sort	0	00:00:01.36	00:00:06.12
Pass 2	138	00:00:02.25	00:00:10.15
Symbol table output	9	00:00:00.07	00:00:00.65
Psect synopsis output	5	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	657	00:00:13.37	00:00:54.30

The working set limit was 1350 pages.
51439 bytes (101 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 852 non-local and 31 local symbols.
689 source lines were read in Pass 1, producing 14 object records in Pass 2.
21 pages of virtual memory were used to define 20 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	10
_255DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	17

951 GETS were required to define 17 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:ASTDEL/OBJ=OBJ$:ASTDEL MSRC$:ASTDEL/UPDATE=(ENH$:ASTDEL)+EXECML$/LIB

ALLOCPFN
LIS

ACCOUNT
LIS

CMODSSDSP
MAR

SYSMAR
MAR

SYSPARAM
MAR

ASTDEL
LIS

0373 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CMODSSDSP
LIS

BUGCHECK
LIS

COMDRVSUB
LIS

CLUSTRVEC
LIS

BUFFERCTL
LIS

DEADLOCK
LIS

CVTFILNAM
LIS

BOOPARAM
LIS

CJFSYSVEC
LIS

CVTATB
LIS

BUGCHKMSG
LIS

CONSOLIO
LIS