Educational Services

digital™

VMS Internals I: Mechanisms and Overview Source Listings Book

EY-9768E-DA-0002

October 1989

CONTENTS

1 ASTDEL.LIS

ASTDEL - AST ENQUEUE AND DELIVERY 10-MAY-1989 15:46:14 VAX MACRO V5.0-8 Page 0
Table of contents

```
ASTDEL - AST ENQUEUE AND DELIVERY 10-MAY-1989 15:46:14 VAX MACRO V5.0-8 Page 1
X-13 12-APR-1988 11:04:19 [SYS.SRC]ASTDEL.MAR;1 (1)

     1          .TITLE  ASTDEL - AST ENQUEUE AND DELIVERY
     2          .IDENT  'X-13'
     3
     4 ;
     5 ;*****************************************************************************
     6 ;*                                                                           *
     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
     9 ;*  ALL RIGHTS RESERVED.                                                     *
    10 ;*                                                                           *
    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
    16 ;*  TRANSFERRED.                                                             *
    17 ;*                                                                           *
    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
    20 ;*  CORPORATION.                                                             *
    21 ;*                                                                           *
    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
    24 ;*                                                                           *
    25 ;*                                                                           *
    26 ;*****************************************************************************
    27
    28 ;++
    29 ; FACILITY: EXECUTIVE, SCHEDULER
    30 ;
    31 ; ABSTRACT:
    32 ;       ASTDEL CONTAINS THE AST DELIVERY INTERRUPT SERVICE ROUTINE AND THE
    33 ;       ASSOCIATED SUBROUTINES G^SCH$QAST AND G^SCH$NEWLVL.  THESE ROUTINES
    34 ;       IMPLEMENT THE PRIMITIVE AST QUEUEING AND DELIVERY MECHANISMS.
    35 ;
    36 ; ENVIRONMENT:
    37 ;       MODE = KERNEL
    38 ;--
    39 ;
    40 ;       .PAGE
    41         .SBTTL  HISTORY                 ; DETAILED
    42 ;
    43 ; AUTHOR:       R. HUSTVEDT      CREATION DATE: 1-SEP-76
    44 ;
    45 ; MODIFIED BY:
    46 ;
    47 ;       X-13    SSA0002         Stan Amway              12-Apr-1988
    48 ;               In SCH$QAST, don't report AST event if process is
    49 ;               current on any CPU.
    50 ;
    51 ;       X-12    SSA0001         Stan Amway              25-Nov-1987
    52 ;               Prevent ACCVIO when referencing CTL$GB_SOFT_AST_DISABLE.
    53 ;               This can happen because the cell is located on a user
    54 ;               writable, pageable page that will be deleted during
    55 ;               process deletion.
    56 ;
    57 ;       X-11    RAB0004         Richard A. Bishop       17-Nov-1987
```

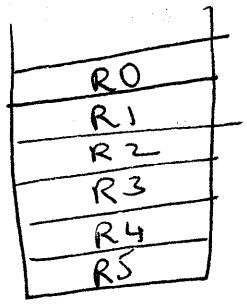
```
;                Add PRESERVE=NO to LOCK macro in SCH$QAST by moving
;                one MOVZWL.
;
;        X-10    JLV5001         Jake VanNoy             27-Aug-1987
;                Add soft-disable of user mode ASTs.  A bit in P1
;                space now functions as if $SETAST had been called.
;
;        X-9     RNG5009         Rod Gamache             26-Aug-1987
;                Fix register setup from last edit.
;
;        X-8     RNG5008         Rod Gamache             23-Jul-1987
;                Add some performance enhancements.
;
;        X-7     RNG5007         Rod Gamache             3-Mar-1987
;                Make SCB dispatch directly to SCH$ASTDEL in this module.
;
;        X-6     RNG5006         Rod Gamache             27-Feb-1987
;                Remove MPH$xxx ASMP multiprocessing hooks.
;
;        X-5     SF04002         Stephen Fiorelli        05-Feb-1987
;                Change reference to sch$gl_nullpcb to sch$ar_nullpcb.
;
;        X-4     WCT0015         Ward C. Travis           6-Jan-1987
;                Update  outdated  SMPLOCK,  SMPUNLOCK uses  to  LOCK,
;                UNLOCK for SMP.
;
;        X-2     SF04001         Stephen Fiorelli        23-Apr-1986
;                exe$astret no longer a universal symbol.
;
;        V03-003 SRB0137         Steve Beckhardt         12-Jul-1984
;                Fixed bug where ACBs queued to non-existent processes
;                are deallocated without checking the NODELETE bit.
;
;        V03-002 WMC0001         Wayne Cardoza           29-Feb-1984
;                Optimize queuing to non-resident process.
;
;        V03-001 ACG0341         Andrew C. Goldstein,    10-Jun-1983  18:59
;                Check and correct stack pointer before AST delivery
;                to mask 780 microcode bug in RET instruction.
;

```

```
                        .SBTTL  DECLARATIONS

;
; INCLUDE FILES:
;
                        $ACBDEF                         ; AST CONTROL BLOCK DEFINITIONS
                        $CPUDEF                         ; PER-CPU DATA BLOCK DEFINITIONS
                        $IPLDEF                         ; IPL DEFINITIONS
                        $PCBDEF                         ; PCB DEFINITIONS
                        $PHDDEF                         ; PHD DEFINITIONS
                        $PRDEF                          ; PROCESSOR REGISTER DEFINITIONS
                        $PRIDEF                         ; PRIORITY INCREMENT CLASS DEFS
                        $PSLDEF                         ; PSL FIELD DEFINITIONS
                        $SSDEF                          ; STATUS CODE DEFINITIONS
                        $STATEDEF                       ; DEFINE PCB STATES

;
; EQUATED SYMBOLS:
;
ASTEXIT=0                                               ; AST EXIT CHANGE MODE CODE
;
; OWN STORAGE:
;
```

```
ASTDEL - AST ENQUEUE AND DELIVERY 10-MAY-1989 15:46:14 VAX MACRO V5.0-8 Page 4
X-13 Initialization Routine 12-APR-1988 11:04:19 [SYS.SRC]ASTDEL.MAR;1 (3)

  124          .SBTTL  Initialization Routine
  125 ;
  126 ; ASTDEL_INIT
  127 ;
  128 ;        Initialize the SCB with the ASTDEL handler address.
  129 ;
  130 ;
  131          DECLARE_PSECT EXEC$INIT_CODE
  132
  133 INITIALIZATION_ROUTINE -
  134          ASTDEL_INIT
  135
  136 ASTDEL_INIT:
  137          MOVL    G^EXE$GL_SCB,R0                 ; SCB address
  138          MOVAB   W^SCH$ASTDEL,^X88(R0)           ; Fill in ASTDEL vector
  139          MOVL    #SS$_NORMAL,R0
  140          RSB
  141
  142
  143          DECLARE_PSECT   EXEC$NONPAGED_CODE
```

```
ASTDEL - AST ENQUEUE AND DELIVERY 10-MAY-1989 15:46:14 VAX MACRO V5.0-8 Page 5
X-13 SCH$ASTDEL - AST DELIVERY INTERRUPT HAND 12-APR-1988 11:04:19 [SYS.SRC]ASTDEL.MAR;1 (4)

  145          .SBTTL  SCH$ASTDEL - AST DELIVERY INTERRUPT HANDLER
  146
  147 ;++
  148 ; FUNCTIONAL DESCRIPTION:
  149 ;       SCH$ASTDEL RECEIVES THE AST DELIVERY INTERRUPT (IPL - 2) WHICH
  150 ;       IS INITIATED BY AN REI INSTRUCTION DETECTING ASTLVL LESS THAN
  151 ;       OR EQUAL TO PSL<CURRENT_MODE>.  THE HEAD OF THE AST QUEUE
  152 ;       FOR THE CURRENT PROCESS IS REMOVED AND PROCESSED.  SPECIAL
  153 ;       KERNEL MODE ASTS ARE PROCESSED WITH IPL REMAINING AT IPL 2.
  154 ;       NORMAL ASTS ARE DELIVERED BY PUSHING THE AST INFORMATION ON
  155 ;       THE STACK OF THE MODE RECEIVING THE AST AND THE AST ACTIVE
  156 ;       BIT FOR THAT MODE IS SET TO PREVENT SUBSEQUENT ASTS UNTIL THE
  157 ;       CURRENT ONE FOR THAT MODE HAS BEEN PROCESSED.
  158 ;       SPURIOUS AST INTERRUPTS WILL BE DETECTED AND IGNORED.
  159 ;
  160 ;
  161 ; CALLING SEQUENCE:
  162 ;       IPL - 2  INTERRUPT
  163 ;
  164 ;
  165 ; INPUT PARAMETERS:
  166 ;       00(SP) = PC AT AST DELIVERY INTERRUPT
  167 ;       04(SP) = PSL AT AST DELIVERY INTERRUPT
  168 ;
  169 ; IMPLICIT INPUTS:
  170 ;       PCB OF CURRENT PROCESS LOCATED VIA CPU$L_CURPCB IN PER-
  171 ;       CPU DATA BLOCK FOR THIS CPU.
  172 ;       AST CONTROL BLOCK AT HEAD OF AST QUEUE FOR PROCESS
  173 ;
  174 ; OUTPUT PARAMETERS:
  175 ;       NONE
  176 ;
  177 ; COMPLETION CODES:
  178 ;       NONE
  179 ;--
  180
  181          .ALIGN  LONG                    ; INTERRUPT ROUTINES ON LW BOUND
  182          UNIVERSAL_SYMBOL        SCH$ASTDEL
  183 ;SCH$ASTDEL::                            ; AST DELIVERY INTERRUPT HANDLER
  184          PUSHL   R5                      ; SAVE R0-R5
  185          PUSHL   R4                      ; ONE REGISTER AT A TIME FOR
  186          PUSHL   R3                      ; SPEED AT THE
  187          PUSHL   R2                      ; EXPENSE OF SPACE AND
  188          PUSHL   R1                      ; CLARITY
  189          PUSHL   R0                      ; (PUSHLS ARE FASTER THAN MOVQ OR PUSHR)
  190 GETNEXT:CLRL     R2                      ; ASSUME NOT SOFT DISABLE
  191          MOVL    G^CTL$GL_PCB,R4         ; GET POINTER TO CURRENT PCB
  192          BBS     #PCB$V_DELPEN,-         ; If delete pending for process,
  193                  PCB$L_STS(R4),10$       ; skip fetch of soft AST disable flag
  194          TSTL    SP                      ; TEST SP
  195          BLSS    10$                     ; BRANCH IF SWAPPER PROCESS; STACK IN S0
  196          MOVZBL  G^CTL$GB_SOFT_AST_DISABLE,R2 ; MUST FETCH BEFORE RAISING IPL
  197 10$:     LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
  198                  LOCKIPL=#IPL$_SYNCH,-   ; RAISE IPL
  199                  PRESERVE=NO             ; DON'T PRESERVE R0
  200          REMQUE  @PCB$L_ASTQFL(R4),R5    ; AND REMOVE HEAD OF QUEUE
  201          BVS     QEMPTY                  ; EXIT IF QUEUE EMPTY
```

```
ASTDEL - AST ENQUEUE AND DELIVERY 10-MAY-1989 15:46:14 VAX MACRO V5.0-8 Page 6
X-13 SCH$ASTDEL - AST DELIVERY INTERRUPT HAND 12-APR-1988 11:04:19 [SYS.SRC]ASTDEL.MAR;1 (4)

  202          BBCC    #ACB$V_KAST,ACB$B_RMOD(R5),NORM ; BR IF NORMAL AST
```



```
ASTDEL - AST ENQUEUE AND DELIVERY 10-MAY-1989 15:46:14 VAX MACRO V5.0-8 Page 7
X-13 KAST - SPECIAL KERNEL AST DISPATCHING 12-APR-1988 11:04:19 [SYS.SRC]ASTDEL.MAR;1 (5)

                .SBTTL  KAST - SPECIAL KERNEL AST DISPATCHING
;
; AST LEVEL WILL BE LEFT AT 0 (KERNEL) WHILE PROCESSING THE SPECIAL KERNEL
; AST.  IT WILL BE CORRECTED EVENTUALLY BEFORE IPL IS DROPPED BELOW IPL$_ASTDEL
; BY REPEATED TRIPS TO GETNEXT.
;
; THE KERNEL AST ROUTINE IS ENTERED VIA A JSB TO THE SPECIFIED
; ADDRESS WITH IPL=2 AND THE POINTER TO THE AST CONTROL BLOCK
; IN R5.  IT IS THE RESPONSIBILITY OF THE KERNEL AST ROUTINE
; TO PROPERLY RELEASE OR OTHERWISE DISPOSE OF THE AST CONTROL
; BLOCK.  THE PCB BASE ADDRESS IS IN R4.
;
; REGISTERS R0-R5 HAVE BEEN PRESERVED AND ARE AVAILABLE FOR
; USE BY THE AST ROUTINE.
;
;
; SINCE KAST ROUTINES OFTEN QUEUE NORMAL ASTS, QIO FOR EXAMPLE, ATTEMPT
; TO DELIVER FURTHER ASTS WITHOUT INCURRING REDUNDANT EXIT AND ENTRY COSTS.
;

        UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATBASE
                NEWIPL=#IPL$_ASTDEL,-   ; DROP IPL
                PRESERVE=NO
        PUSHAB  GETNEXT                 ; SET RETURN ADDRESS TO CONTINUE WITH Q
        JMP     @ACB$L_KAST(R5)         ; DO KERNEL MODE AST

;
; THIS UNUSUAL CALLING SEQUENCE IS TO MINIMIZE THE NUMBER OF TAKEN BRANCHES
; AND IS EQUIVALENT TO:
;       JSB     @ACB$L_KAST(R5)
;       BRB     GETNEXT
;
```

```
                        .SBTTL  ASTDEL EXITS
;
; THE CTL$GB_SOFT_AST_DISABLE LOW BIT WAS SET.  CLEAR THE AST ENABLE BIT, JUST
; AS IF $SETAST HAD BEEN CALLED.  ALSO, SET THE BIT THAT TELLS THE USER
; LEVEL CODE TO CALL $SETAST TO REENABLE ASTS WHEN DONE.
;

SOFT_AST_DISABLE:

        BBCC    R3,PCB$B_ASTEN(R4),10$    ; REALLY DISABLE ASTS
10$:    MOVB    #1,G^CTL$GB_REENABLE_ASTS ; FLAG FOR USER THREAD
;
; IF THE ASTMODE IS DISABLED OR ACTIVE, THEN SET ASTLVL TO PREVENT
; FURTHER INTERRUPTS.  THERE IS AN ASSUMPTION (AND HAS BEEN FOREVER)
; THAT AN INNER ACCESS MODE WILL NOT EXIT TO AN OUTER ACCESS MODE
; WITH EITHER AN ACTIVE AST OR LEAVING ASTS DISABLED.
;

BLOCKED:                                ;

;
; THE AST DELIVERY INTERRUPT WAS SPURIOUS, A PENDING IPL-2 INTERRUPT LEFT
; OVER FROM THE PREVIOUS PROCESS.  THESE OCCUR INFREQUENTLY AND ARE DETECTED
; BY COMPARING THE ACCESS MODE OF THE FIRST AST IN THE QUEUE WITH THE CURRENT
; MODE OF THE INTERRUPTED PSL.  SPURIOUS IPL-2 INTERRUPTS ARE ALSO DETECTED
; BY THE REMQUE ABOVE FINDING AN EMPTY QUEUE.
;

SPURIOUS:                               ;
        INSQUE  (R5),PCB$L_ASTQFL(R4)   ; REQUEUE AT HEAD OF QUEUE
        BRB     SETLVL                  ; AND SET NEW ASTLEVEL

;
; THE AST QUEUE IS NOW EMPTY.   EITHER THE AST DELIVERY INTERRUPT IS SPURIOUS
; OR ALL OF THE QUEUED ASTS HAVE BEEN CANCELED BY SIMPLY REMOVING THEM FROM THE
; QUEUE.  INSURE THAT ASTLVL IS SET TO PREVENT FURTHER INTERRUPTS.
;
; R4 - PCB ADDRESS
;
QEMPTY:
        MOVL    #4,R1                   ; SET NULL AST LEVEL
;
; SET AST LEVEL TO BEST ESTIMATE OF CORRECT ACCESS MODE.
;
; R1 - NEW ACCESS MODE TO SET IN ASTLVL
;
SETLVL: MOVL    PCB$L_PHD(R4),R0        ; GET PROCESS HEADER ADDRESS
        MOVB    R1,PHD$B_ASTLVL(R0)     ; SET ASTLEVEL IN HW PCB
        MTPR    R1,#PR$_ASTLVL          ; AND PROCESSOR REGISTER
ASTDEXIT:                               ; AST DELIVERY EXIT
        UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATABASE/NO IPL CHANGE
                PRESERVE=NO             ; DON'T PRESERVE R0
        MOVQ    (SP)+,R0                ; RESTORE REGISTERS R0,R1
        MOVQ    (SP)+,R2                ; RESTORE REGISTERS R2,R3
        MOVQ    (SP)+,R4                ; RESTORE REGISTERS R4,R5
        REIMAC                          ; AND RETURN
```

```
ASTDEL - AST ENQUEUE AND DELIVERY 10-MAY-1989 15:46:14 VAX MACRO V5.0-8 Page 9
X-13 NORM - NORMAL AST DELIVERY 12-APR-1988 11:04:19 [SYS.SRC]ASTDEL.MAR;1 (7)

  294          .SBTTL  NORM - NORMAL AST DELIVERY
  295 ;
  296 ;        AT THIS POINT THE KERNEL STACK IS:
  297 ;        00(SP) = SAVED R0
  298 ;        04(SP) = SAVED R1
  299 ;        08(SP) = SAVED R2
  300 ;        12(SP) = SAVED R3
  301 ;        16(SP) = SAVED R4
  302 ;        20(SP) = SAVED R5
  303 ;        24(SP) = SAVED PC
  304 ;        28(SP) = SAVED PSL
  305 ;
  306 ;        R0 - PHD ADDRESS
  307 ;        R1 - ACCESS MODE OF NEXT ACB OR NULL AST LEVEL
  308 ;        R2 - COPY OF CTL$GB_SOFT_AST_DISABLE
  309 ;        R4 - PCB ADDRESS
  310 ;        R5 - ACB ADDRESS
  311 ;
  312
  313 NORM:                                       ; NORMAL AST DELIVERY
  314          ASSUME  ACB$V_MODE EQ 0
  315          ASSUME  ACB$S_MODE EQ 2
  316          CLRL    R1                         ; BACKGROUND R3 WITH ZEROES
  317          BICB3   #^C<3>,ACB$B_RMOD(R5),R1; EXTRACT ACCESS MODE FOR CURRENT AST
  318          CMPZV   #PSL$V_CURMOD,#PSL$S_CURMOD,28(SP),R1  ; IS CURRENT MODE LEGAL
  319          BLSS    SPURIOUS                   ; BR IF SPURIOUS
  320          MOVAB   (R1)+,R3                   ; SET FOR NEXT ACCESS MODE
  321          BBC     R3,PCB$B_ASTEN(R4),BLOCKED ; BR IF AST DISABLED
  322          CMPB    #PSL$C_USER,R3             ; USER MODE?
  323          BNEQ    20$                        ; BR IF NOT
  324          BLBS    R2,SOFT_AST_DISABLE        ; BRANCH IF SOFT-DISABLED
  325 20$:     BBSS    R3,PCB$B_ASTACT(R4),BLOCKED ; SET AST ACTIVE
  326          MTPR    R1,#PR$_ASTLVL             ; SET AST LEVEL IN PROCESSOR REGISTER
  327          MOVL    PCB$L_PHD(R4),R0           ; GET PROCESS HEADER ADDRESS
  328          MOVB    R1,PHD$B_ASTLVL(R0)        ; SET AST LEVEL IN HW PCB
  329          BBC     #ACB$V_QUOTA,ACB$B_RMOD(R5),30$ ; SKIP IF NO QUOTA ACCOUNTING
  330          ADAWI   #1,PCB$W_ASTCNT(R4)        ; UPDATE OUTSTANDING COUNT
  331 30$:                                        ; AND DELIVER AST
  332          UNLOCK  LOCKNAME=SCHED,-           ; UNLOCK SCHED DATABASE
  333                  NEWIPL=#IPL$_ASTDEL,-      ; DROP IPL
  334                  PRESERVE=NO                ; DON'T PRESERVE R0
  335 ;
  336 ;        A NEW VALUE FOR ASTLVL HAS NOW BEEN COMPUTED AND SET.
  337 ;        THE AST REPRESENTED BY THE AST CONTROL BLOCK LOCATED VIA
  338 ;        R5 CAN NOW BE DELIVERED.
  339 ;
  340          TSTL    R3                         ; CHECK FOR DELIVERY TO KERNEL
  341          BNEQ    NOTKMODE                   ; BR IF NOT KERNEL MODE
```

```
343          .SBTTL  NORMAL KERNEL MODE AST
344 ;
345 ;        DELIVER NORMAL AST TO KERNEL MODE
346 ;
347 KMODE:   MOVQ    (SP)+,R0                ; RESTORE R0,R1
348          MOVAL   W^ASTEXIT_CHMK,-(SP)    ; GET ADDRESS OF EXIT
349          CMPL    20(SP),(SP)+            ; DELIVERY OCCUR DURING ASTEXIT REI?
350          BNEQ    40$                     ; NO, JUST CONTINUE
351          BITB    #<PSL$M_CURMOD-         ; WAS INTERRUPTED MODE KERNEL?
352                  @<-PSL$V_CURMOD>>,23(SP);
353          BEQL    50$                     ; YES, PREVIOUS AST R0,R1,PC,PSL ALREADY ON
354 ;
355 ;        00(SP) = R2, 04(SP) = R3, 08(SP) = R4, 12(SP) = R5,
356 ;        16(SP) = PC, 20(SP) = PSL
357 ;
358 40$:
359          MOVQ    8(SP),-(SP)             ; SHUFFLE STACK
360 ;
361 ;        00(SP) = R4, 04(SP) = R5, 08(SP) = R2, 12(SP) = R3,
362 ;        16(SP) = R4, 20(SP) = R5, 24(SP) = PC, 28(SP) = PSL
363 ;
364          MOVQ    8(SP),-(SP)             ; OPEN FOR AST ARG LIST
365 ;
366 ;        00(SP) = R2, 04(SP) = R3, 08(SP) = R4, 12(SP) = R5,
367 ;        16(SP) = R2, 20(SP) = R3, 24(SP) = R4, 28(SP) = R5,
368 ;        32(SP) = PC, 36(SP) = PSL
369 ;
370          MOVQ    R0,24(SP)               ; SET R0,R1 IN ARG LIST
371 50$:    MOVL    ACB$L_ASTPRM(R5),20(SP) ; SET AST PARAMETER IN ARG LIST
372          MOVL    #5,16(SP)               ; SET COUNT FOR ARGUMENT LIST
373          MOVL    R5,R0                   ; RELEASE AST CONTROL BLOCK
374          PUSHL   ACB$L_AST(R5)           ; SAVE AST ROUTINE ADDRESS
375          BBC     #ACB$V_PKAST,ACB$B_RMOD(R5),60$ ; BR IF NO PIGGY-BACK KAST
376 ;
377 ; CALL PIGGY-BACK SPECIAL KERNEL AST ROUTINE.
378 ;        R5 - ACB ADDRESS (MUST BE PRESERVED)
379 ;        IPL = IPL$_ASTDEL (MUST NOT BE LOWERED)
380 ;
381          JSB     @ACB$L_KAST(R5)         ; CALL KAST ROUTINE
382          BRB     70$                     ; NO DELETE FOR PKAST
383 60$:    BBS     #ACB$V_NODELETE,ACB$B_RMOD(R5),70$; BR IF NOT DELETEABLE
384          JSB     G^EXE$DEANONPAGED               ; TO DYNAMIC POOL
385 70$:    MOVQ    (SP)+,R1                ; RESTORE R1,R2
386          MOVQ    (SP)+,R3                ; RESTORE R3,R4
387          MOVL    (SP)+,R5                ; RESTORE R5
388          SETIPL  #0                      ; DROP IPL TO ZERO
389 ;        BRB     EXE$ASTDEL              ; FALL THROUGH TO CALL AST ROUTINE
```

```
391 ;
392 ;       CALL AST ROUTINE WITH AST ARGUMENT LIST
393 ;
394 ;       THE CALL IS EXECUTED AT THE MODE WHICH RECEIVED THE AST WITH
395 ;       THE AST ARGUMENT LIST ON THE TOP OF THE STACK.  WHEN THE
396 ;       AST ROUTINE RETURNS FROM THE CALL, AN ASTEXIT CHANGE MODE
397 ;       TO KERNEL INSTRUCTION WILL BE ISSUED.  ASTEXIT WILL RESET
398 ;       THE AST ACTIVE BIT FOR THE CURRENT MODE AND MAY CAUSE DELIVERY
399 ;       OF ADDITIONAL ASTS.
400 ;
401 ;       AST ARGUMENT LIST:
402 ;       ------------------
403 ;
404 ;       00(SP) = NUMBER OF ARGUMENTS, =5
405 ;       04(SP) = AST PARAMETER
406 ;       08(SP) = SAVED R0
407 ;       12(SP) = SAVED R1
408 ;       16(SP) = SAVED PC
409 ;       20(SP) = SAVED PSL
410 ;
411         UNIVERSAL_SYMBOL        EXE$ASTDEL
412 ;EXE$ASTDEL::                            ; DELIVER AST CALL
413         CALLG   (SP),(R1)                ; CALL AST ROUTINE
414
415 EXE$ASTRET::                            ; RETURN ADDRESS FOR AST CALL
416         ADDL    #8,SP                    ; REMOVE ARG COUNT AND ASTPRM
417         CHMK    S^#ASTEXIT               ; AND EXIT FROM AST ROUTINE
418 ASTEXIT_CHMK:                           ; RETURN ADDRESS FOR AST EXIT CHMK
419         MOVQ    (SP)+,R0                 ; RESTORE R0,R1
420         REIMAC                           ; EXECUTE REI IN MODE OF AST
421
```

```
                .SBTTL  NORMAL EXEC, SUPER AND USER MODE AST

                .ENABLE LSB
;
; SPECIAL CASE CODE TO HANDLE THE SITUATION WHERE THE OUTER MODE SP HAS
; BEEN FOUND TO BE BELOW THE FP, MEANING THE STACK IS CORRUPTED. THIS
; SITUATION CAN RESULT FROM A 780 MICROCODE BUG SHICH LEAVES THE SP
; WRONG WHEN A RET PAGEFAULTS WHILE LOOKING AT THE ARG LIST. WE CHECK
; FOR THIS CASE AND SET SP = FP, WHICH IS ADEQUATE TO SERVICE THE AST
; AND LET THE RET RE-EXECUTE.
;
1$:             ADDL3   #18*4,FP,R0             ; RANGE LIMIT IS MAXIMAL FRAME + ALIGNMENT
                CMPL    R1,R0                   ; CHECK IF SP IS WITHIN RANGE
                BGTRU   5$                      ; IF NOT, SOME OTHER PROBLEM - DON'T FIX
                MOVL    FP,R1                   ; SET OUTER MODE SP = FP
                BRB     5$                      ; AND CONTINUE

;
;       DELIVER NORMAL AST FOR EXEC, SUPER AND USER MODE
;

NOTKMODE:                                       ; NOT AN AST FOR KERNEL MODE
;
                MFPR    R3,R1                   ; GET STACK POINTER
                CMPL    R1,FP                   ; SEE IF STACK POINTER IS ABOVE FP
                BGTRU   1$                      ; BRANCH IF NOT - POSSIBLE FIXUP
5$:             IFNOWRT #24,-24(R1),STACKERR,R3 ; ENOUGH STACK SPACE??
                MOVQ    24(SP),-(R1)            ; MOVE PC,PSL TO PROPER STACK
                MOVQ    (SP)+,-(R1)             ; AND R0,R1 FROM KERNEL STACK
                MOVAL   W^ASTEXIT_CHMK,-(SP)    ; GET ADDRESS OF EXIT CODE
                CMPL    20(SP),(SP)+            ; DELIVERY OCCUR DURING ASTEXIT REI?
                BEQL    50$                     ; YES, CHECK FOR SAME MODE
10$:            MOVL    ACB$L_ASTPRM(R5),-(R1)  ; SET AST PARAMETER IN ARG LIST
                MOVL    #5,-(R1)                ; AND FINALLY, ARGUMENT COUNT OF 5
                MTPR    R1,R3                   ; SAVE UPDATED STACK POINTER
                PUSHL   ACB$L_AST(R5)           ; STACK AST ENTRY POINT
                MOVL    R5,R0                   ; SET ADDRESS OF ACB FOR RELEASE
                MOVAB   G^EXE$ASTDEL,20(SP)     ; SET PC TO AST DELIVERY CALL
                MOVAL   (R3)[R3],R3             ; MODE=MODE*5, CURMOD=PRVMOD
                ASSUME  PSL$V_CURMOD EQ PSL$V_PRVMOD+2; FOR ABOVE MOVAL
                ASHL    #PSL$V_PRVMOD,R3,24(SP) ; SYNTHESIZE PSL FOR PROPER MODE
                BITB    #<ACB$M_NODELETE!ACB$M_PKAST>,ACB$B_RMOD(R5) ; SPECIAL ACTIONS?
                BNEQ    40$                     ; BR IF SO AND DECODE
20$:            JSB     G^EXE$DEANONPAGED               ; RELEASE AST CONTROL BLOCK
30$:            MOVQ    (SP)+,R1                ; RESTORE R1,R2
                MOVQ    (SP)+,R3                ; RESTORE R3,R4
                MOVL    (SP)+,R5                ; RESTORE R5
                REIMAC                          ; AND ENTER AST MODE
                                                ; DROPS IPL TO ZERO
40$:            BBC     #ACB$V_PKAST,ACB$B_RMOD(R5),30$ ; BR IF NO PIGGY-BACK KAST
;
; CALL PIGGY-BACK SPECIAL KERNEL AST ROUTINE.
;       R5 - ACB ADDRESS (MUST BE PRESERVED)
;       IPL = IPL$_ASTDEL (MUST NOT BE LOWERED)
;
                JSB     @ACB$L_KAST(R5)         ; CALL KAST ROUTINE
                BRB     30$                     ; NO DELETE FOR PKAST
```

```
ASTDEL - AST ENQUEUE AND DELIVERY 10-MAY-1989 15:46:14 VAX MACRO V5.0-8 Page 13
X-13 NORMAL EXEC, SUPER AND USER MODE AST 12-APR-1988 11:04:19 [SYS.SRC]ASTDEL.MAR;1 (10)

;
; SPECIAL CASE FOR AST DURING AST EXIT.
;
50$:    CMPZV   #PSL$V_CURMOD,-         ; WAS AST'S MODE THE ONE INTERRUPTED?
                #PSL$S_CURMOD,20(SP),R3 ;
        BNEQ    10$                     ; NO, JUST CONTINUE
        MFPR    R3,R1                   ; RESET SP - R0,R1,PC,PSL ALREADY ON STACK
        BRB     10$                     ; CONTINUE

        .DISABLE LSB

;
;       REFLECT STACK ERROR
;
STACKERR:                               ; ERROR IN STACK MOVE
        CMPL    #PSL$C_USER,R3          ; IS THIS AST FOR USER MODE?
        BNEQ    10$                     ; NO, THEN WE CANT EXTEND THE STACK
        PUSHR   #^M<R1,R2,R3,R4,R5>     ; SAVE NECESSARY REGISTERS
        MOVAB   -24(R1),R2              ; COMPUTE DESIRED STACK TOP ADDRESS
        JSB     G^EXE$EXPANDSTK         ; EXPAND USER STACK TO DESIRED SIZE
        POPR    #^M<R1,R2,R3,R4,R5>     ; RESTORE REGISTERS
        BLBS    R0,99$                  ; CONTINUE IF SPACE CREATED
10$:    BBCC    R3,PCB$B_ASTACT(R4),20$ ; CLEAR AST ACTIVE BIT
20$:    MOVQ    16(SP),-(SP)            ; CREATE SPACE ON STACK
        MOVQ    16(SP),-(SP)            ; BY MOVING R2-R5 DOWN
        MOVQ    40(SP),32(SP)           ; SAVE PC,PSL AT INTERRUPT
        MOVL    ACB$L_AST(R5),40(SP)    ; SET PC AT FAULT TO AST ADDRESS
        MOVL    ACB$L_ASTPRM(R5),28(SP) ; SET ASTPRM IN ARGLIST
        MOVL    R1,24(SP)               ; SAVE STACK VA AT FAULT
        MULL    #1+<1@<PSL$V_CURMOD-PSL$V_PRVMOD>>,R3   ; CURRENT MODE = PREV
        ROTL    #PSL$V_PRVMOD,R3,44(SP) ; SYNTHESIZE NEW PSL FOR FAULT
        MOVL    R5,R0                   ; SET ADDRESS FOR RELEASE OF ACB
        BBC     #ACB$V_PKAST,ACB$B_RMOD(R5),30$ ; BR IF NO PIGGY-BACK KAST
;
; CALL PIGGY-BACK SPECIAL KERNEL AST ROUTINE.
;       R5 - ACB ADDRESS (MUST BE PRESERVED)
;       IPL = IPL$_ASTDEL (MUST NOT BE LOWERED)
;
        JSB     @ACB$L_KAST(R5)         ; CALL KAST ROUTINE
        BRB     40$                     ; NO DELETE FOR PKAST
30$:    BBS     #ACB$V_NODELETE,ACB$B_RMOD(R5),40$; BR IF NOT DELETEABLE
        JSB     G^EXE$DEANONPAGED               ; AND DEALLOCATE IT
40$:    POPR    #^M<R2,R3,R4,R5>        ; RESTORE ALL REGISTERS
        POPR    #^M<R0,R1>              ; FROM POINT OF INTERRUPT
        SETIPL  #0                      ; DROP IPL TO 0
        JMP     G^EXE$ASTFLT            ; REFLECT EXCEPTION

99$:    BRW     NOTKMODE                ; LONG BRANCH
```

```
  529          .SBTTL  EXE$IPAPBKAST - SPECIAL PIGGY BACK KAST FOR IPAST SERVICE
  530 ;++
  531 ; FUNCTIONAL DESCRIPTION:
  532 ;        THIS ROUTINE IS A SPECIAL PIGGY BACK KAST ROUTINE CALLED BY
  533 ;        ASTDEL DURING DELIVERY OF THE INTER-PROCESS AST.  IT TAKES THE
  534 ;        IPAST INDEX RESIDING IN ACB$L_AST AND USES IT TO INDEX INTO THE
  535 ;        VECTOR OF AST ADDRESSES LOCATED IN THE VECTOR PAGE OF THE CONTROL
  536 ;        REGION.  THIS ADDRESS REPLACES WHAT WAS ON THE STACK WHEN THIS
  537 ;        ROUTINE WAS ENTERED.  ROUTINE IS INCLUDED HERE SINCE ITS HAS
  538 ;        KNOWLEDGE OF HOW STACK LOOKS WHEN PIGGYBACK AST IS CALLED.
  539 ;
  540 ; INPUTS:
  541 ;        4(SP) --> AST ADDRESS
  542 ;        ACB$L_AST(R5) = INDEX OF IPASTS
  543 ;
  544 ; OUTPUTS:
  545 ;        4(SP) --> NEW AST ADDRESS
  546 ;
  547 ;--
  548
  549          UNIVERSAL_SYMBOL        EXE$IPAPBKAST
  550 ;EXE$IPAPBKAST::
  551          MOVL    ACB$L_AST(R5),R2        ; Pick up index
  552          MOVL    G^CTL$AL_IPASTVEC[R2],R0; Get IPAST address
  553          BEQL    10$                     ; No longer in use
  554          MOVL    R0,4(SP)                ; Fixup AST delivery address
  555 10$:     MOVL    R5,R0                   ; Transfer address of packet
  556          JMP     G^EXE$DEANONPAGED       ; Get rid of it
```

```
558          .SBTTL  SCH$QAST - ENQUEUE AST CONTROL BLOCK FOR PROCESS
559 ;++
560 ; FUNCTIONAL DESCRIPTION:
561 ;       G^SCH$QAST INSERTS THE AST CONTROL BLOCK SUPPLIED IN THE PROPER
562 ;       POSITION BY ACCESS MODE IN THE AST QUEUE OF THE PROCESS SPECIFIED
563 ;       BY THE PID FIELD OF THE AST CONTROL BLOCK.  AN AST ARRIVAL EVENT
564 ;       IS THEN REPORTED FOR THE PROCESS TO REACTIVATE FROM A WAIT STATE
565 ;       IF APPROPRIATE.  THE AST CONTROL BLOCK WILL BE RELEASED IMMEDIATELY
566 ;       IF THE PID SPECIFIES A NON-EXISTENT PROCESS.
567 ;
568 ; CALLING SEQUENCE:
569 ;       BSB/JSB G^SCH$QAST
570 ;
571 ; INPUT PARAMETERS:
572 ;       R2 - PRIORITY INCREMENT CLASS
573 ;       R5 - POINTER TO AST CONTROL BLOCK
574 ;
575 ; IMPLICIT INPUTS:
576 ;       PCB OF PROCESS IDENTIFIED BY PID FIELD
577 ;
578 ; OUTPUT PARAMETERS:
579 ;       R0 - COMPLETION STATUS CODE
580 ;       R4 - PCB ADDRESS OF PROCESS FOR WHICH AST WAS QUEUED
581 ;
582 ; SIDE EFFECTS:
583 ;       THE PROCESS IDENTIFIED BY THE PID IN THE AST CONTROL BLOCK
584 ;       WILL BE MADE EXECUTABLE IF NOT SUSPENDED.
585 ;
586 ; COMPLETION CODES:
587 ;       SS$_NORMAL  -  NORMAL SUCCESSFUL COMPLETION STATUS
588 ;       SS$_NONEXPR -  NON-EXISTENT PROCESS
589 ;--
590          .ENABL  LSB
591 QNONEXPR:
592          ; DEALLOCATE THE ACB AS LONG AS THE NODELETE BIT IS NOT SET.
593          ; THIS REALLY SHOULDN'T HAPPEN, BUT IF IT DOES, WE CHOOSE
594          ; TO POSSIBLY LOSE POOL OVER POSSIBLY CORRUPTING IT.
595
596          BBS     #ACB$V_NODELETE,ACB$B_RMOD(R5),5$; BR IF NOT DELETEABLE
597          MOVL    R5,R0                   ; RELEASE AST CONTROL BLOCK
598          JSB     G^EXE$DEANONPAGED               ; IF NO SUCH PROCESS
599 5$:     MOVZWL  #SS$_NONEXPR,R0         ; SET ERROR STATUS CODE
600          BRW     QEXIT                   ; AND EXIT
601
602          .ALIGN  LONG
603          UNIVERSAL_SYMBOL        SCH$QAST
604 ;SCH$QAST::                             ; ENQUEUE AST FOR PROCESS
605          LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
606                  LOCKIPL=#IPL$_SYNCH,-   ; RAISE IPL
607                  SAVIPL=-(SP),-          ; SAVE CURRENT IPL
608                  PRESERVE=NO
609          MOVZWL  ACB$L_PID(R5),R0        ; GET PROCESS INDEX FOR AST TARGET
610          MOVL    @L^SCH$GL_PCBVEC[R0],R4 ; LOOK UP PCB ADDRESS
611          CMPL    ACB$L_PID(R5),PCB$L_PID(R4) ; CHECK FOR MATCH IN PID
612          BNEQ    QNONEXPR                ; PID MISMATCHES
613          CMPL    G^SCH$AR_NULLPCB,R4     ; QUEUEING TO NULL JOB?
614          BEQL    QNONEXPR                ; DECLARE NON EXISTANT
```

```
615         CLRL    R0                      ; ASSUME KERNEL MODE AND CLEAR HIGH BITS
616         INSQUE  (R5),PCB$L_ASTQFL(R4)   ; ASSUME QUEUE IS EMPTY AND ATTEMPT INSERT
617         BNEQ    50$                     ; BR IF IT WAS NOT EMPTY
618         TSTB    ACB$B_RMOD(R5)          ; CHECK FOR SPECIAL KERNEL AST
619         BLSS    10$                     ; BR IF YES
620         BICB3   #^C<3>,ACB$B_RMOD(R5),R0; GET AST MODE
621 ;
622 ; THE PROCESS HEADER ADDRESS IS ALWAYS A SYSTEM SPACE ADDRESS (NEGATIVE NUMBER)
623 ; WHILE THE PROCESS HEADER IS RESIDENT.  DURING THE OUTSWAP TRANSITION IT IS
624 ; THE BALANCE SLOT INDEX, A SMALL POSITIVE NUMBER.  FINALLY, AFTER OUTSWAP IT
625 ; IS SET TO ZERO.  HENCE, THE FOLLOWING TEST COMBINES THE FETCH OF THE PHD
626 ; ADDRESS WITH THE TEST FOR PROCESS RESIDENCE.
627 ;
628 10$:    MOVL    PCB$L_PHD(R4),R1        ; POINT TO PROCESS HEADER
629         BGEQ    30$                     ; DON'T SET ASTLVL IF NOT RESIDENT
630         MOVB    R0,PHD$B_ASTLVL(R1)     ; SET ASTLVL IN PROCESS HEADER
631         CMPW    #SCH$C_CUR,PCB$W_STATE(R4); IS PROCESS CURRENT ANYWHERE
632         BNEQ    30$                     ; NO THEN REPORT EVENT
633         find_cpu_data   R1              ; GET POINTER TO PER-CPU DATA
634         CMPL    CPU$L_CURPCB(R1),R4     ; IS PROCESS CURRENT ON THIS CPU?
635         BNEQ    40$                     ; BR IF NO
636         MTPR    R0,#PR$_ASTLVL          ; ELSE, SET ASTLVL REGISTER
637         BRB     35$
638 30$:    RPTEVT  AST                     ; REPORT AST ARRIVAL
639 35$:    MOVZWL  #SS$_NORMAL,R0          ; SET SUCCESS STATUS CODE
640 QEXIT:  UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATABASE
641                 NEWIPL=(SP)+,-          ; RESTORE PREVIOUS IPL
642                 CONDITION=RESTORE       ; RESTORE ACCESS COUNT ON LOCK
643         RSB                             ; AND RETURN
644 ;
645 ; IF THE AST IS BEING ENQUEUED FOR THE CURRENT PROCESS, THEN THE REPORTING
646 ; OF THE AST EVENT CAN BE BYPASSED AND THE ASTLVL PROCESSOR REGISTER MUST BE
647 ; SET INSTEAD.
648 ;
649 40$:    MOVL    PCB$L_CPU_ID(R4),R0     ; GET CPU ID THAT PROCESS IS CURRENT ON
650         IPINT_CPU UPDASTLVL             ; REQUEST ASTLVL UPDATE
651         BRB     35$                     ; CONTINUE
652 ;
653 ; THE AST QUEUE WAS NOT EMPTY (ITS USUAL CONDITION) AND THE PROPER
654 ; POSITION FOR THE NEW AST MUST BE LOCATED.  SINCE THE AST CONTROL
655 ; BLOCK HAS BEEN ERRONEOUSLY INSERTED ON THE QUEUE, IT MUST BE REMOVED
656 ; FIRST.
657 ;
658 50$:    REMQUE  (R5),R5                 ; ELSE CORRECT MISTAKE
659         MOVAL   PCB$L_ASTQFL(R4),R1     ; POINT TO QUEUE HEADER
660         MOVL    (R1),R3                 ; GET FIRST ENTRY ON QUEUE
661         TSTB    ACB$B_RMOD(R5)          ; CHECK FOR SPECIAL KERNEL AST
662         BGEQ    70$                     ; BR IF NOT
663 ;
664 ; THE NEW AST IS A SPECIAL KERNEL AST.  IT WILL GO AFTER ALL OTHER SPECIAL
665 ; KERNEL ASTS OR AT THE HEAD OF THE QUEUE IF THERE ARE NONE.
666 ;
667 60$:    CMPL    R1,R3                   ; CHECK FOR END OF QUEUE
668         BEQL    110$                    ; BR IF NOT
669         TSTB    ACB$B_RMOD(R3)          ; CHECK FOR SPECIAL KERNEL IN QUEUE
670         BGEQ    110$                    ; BR IF NOT
671         MOVL    (R3),R3                 ; FLINK ON TO NEXT ACB
```

```
672          BRB     60$                     ;
673 ;
674 ; THE NEW AST IS A NORMAL AST.  IT WILL GO AFTER ALL SPECIAL KERNEL ASTS
675 ; AND ASTS WITH LOWER ACCESS MODE.
676 ;
677 70$:     BICB3   #^C<3>,ACB$B_RMOD(R5),R0; GET AST MODE
678 80$:     CMPL    R1,R3                   ; CHECK FOR END OF QUEUE
679          BEQL    110$                    ; INSERT IF AT END
680          CMPZV   #ACB$V_MODE,#ACB$S_MODE,-
681                  ACB$B_RMOD(R3),R0       ; COMPARE ACCESS MODES
682          BGTR    100$                    ; IF GTR AT RIGHT PLACE
683 90$:     MOVL    (R3),R3                 ; FLINK ON TO NEXT ACB
684          BRB     80$                     ;
685 100$:    TSTB    ACB$B_RMOD(R3)          ; IS THIS ENTRY A SPECIAL KAST?
686          BLSS    90$                     ; YES, MUST GO AFTER THIS
687 ;
688 ; NOW THE CORRECT POSITION HAS BEEN LOCATED.  INSERT THE AST CONTROL BLOCK
689 ; ON THE QUEUE AND COMPUTE THE NEW VALUE FOR ASTLVL BY INTERROGATING THE
690 ; MODE OF THE AST CONTROL BLOCK AT THE HEAD OF THE QUEUE.
691 ;
692 110$:    INSQUE  (R5),@ACB$L_ASTQBL(R3)  ; INSERT AFTER PREVIOUS
693          CLRL    R0                      ; ASSUME KERNEL MODE
694          MOVL    PCB$L_ASTQFL(R4),R1     ; GET HEAD OF AST QUEUE
695          TSTB    ACB$B_RMOD(R1)          ; IS IT KAST?
696          BLSS    120$                    ; BR IF YES TO SET ASTLVL
697          BICB3   #^C<3>,ACB$B_RMOD(R1),R0; GET AST MODE FOR HEAD OF QUEUE
698 120$:    BRW     10$                     ; GO SET ASTLVL
699
700          .DSABL LSB
701          ASSUME  ACB$V_MODE EQ 0
702          ASSUME  ACB$S_MODE EQ 2
703          ASSUME  ACB$V_KAST EQ 7
704
```

```
706          .SBTTL  SCH$NEWLVL - COMPUTE NEW AST LEVEL
707 ;
708 ;        G^SCH$NEWLVL - COMPUTE NEW ASTLVL
709 ;
710 ;        THIS ROUTINE COMPUTES THE NEW AST LEVEL.
711 ;
712 ; INPUT:
713 ;        R4 - ADDRESS OF CURRENT PCB
714 ;
715 ; OUTPUT:
716 ;        PR$_ASTLVL - NEW AST LEVEL
717 ;        PHD$V_ASTLVL - NEW AST LEVEL
718 ;
719          .ENABL  LSB
720          UNIVERSAL_SYMBOL        SCH$NEWLVL
721 ;SCH$NEWLVL::                             ; COMPUTE NEW AST LEVEL
722          CLRL    R2                       ; ASSUME KERNEL MODE
723          LOCK    LOCKNAME=SCHED,-         ; LOCK SCHED DATABASE
724                  LOCKIPL=#IPL$_SYNCH,-    ; RAISE IPL
725                  SAVIPL=-(SP),-           ; SAVE CURRENT IPL
726                  PRESERVE=NO              ; DON'T PRESERVE R0
727          MOVAL   PCB$L_ASTQFL(R4),R0      ; GET ADDRESS OF AST QUEUE
728          MOVL    (R0),R1                  ; GET FLINK
729          CMPL    R0,R1                    ; AND CHECK FOR EMPTY QUEUE
730          BEQL    20$                      ; YES, QUEUE IS EMPTY
731          ASSUME  ACB$V_KAST EQ 7
732          TSTB    ACB$B_RMOD(R1)           ; CHECK FOR KERNEL AST
733          BLSS    10$                      ; BR IF NOT
734          BICB3   #^C<3>,ACB$B_RMOD(R1),R2; GET REQUEST MODE
735 10$:     MOVL    PCB$L_PHD(R4),R0         ; GET ADDRESS OF PHD
736          ASSUME  PHD$S_ASTLVL EQ 8        ; FOR USE OF MOVB
737          MOVB    R2,PHD$B_ASTLVL(R0)      ; SET ASTLVL IN PHD
738          MTPR    R2,#PR$_ASTLVL           ; SET ASTLVL REGISTER
739          UNLOCK  LOCKNAME=SCHED,-         ; UNLOCK SCHED DATABASE
740                  NEWIPL=(SP)+,-           ; RESTORE PREVIOUS IPL
741                  CONDITION=RESTORE,-      ; RESTORE ACCESS COUNT ON LOCK
742                  PRESERVE=NO              ; DON'T PRESERVE R0
743          RSB                              ; RETURN
744 ;
745 20$:     MOVL    #4,R2                    ; SET NULL AST LEVEL
746          BRB     10$                      ;
747          .DSABL  LSB
```

```
749          .SBTTL  G^SCH$SWAPACBS - SWAP AST CONTROL BLOCKS
750          .SBTTL  G^SCH$REMOVACB - REMOVE AST CONTROL BLOCK
751
752 ;++
753 ;
754 ; FUNTIONAL DESCRIPTION:
755 ;
756 ;        G^SCH$SWAPACBS REMOVES AN ACB FROM AN AST QUEUE AND INSERTS ANOTHER
757 ;        ACB IN ITS PLACE.  I.E. IT SWAPS THE TWO ACBS.  THIS IS NORMALLY
758 ;        DONE BECAUSE THE ACB BEING REMOVED IS PART OF A LARGER STRUCTURE
759 ;        THAT MUST BE FREED UP FOR ANOTHER PURPOSE.  THE ACB BEING
760 ;        INSERTED IS NORMALLY A COPY OF THE ONE BEING REMOVED EXCEPT THAT THE
761 ;        ACB$M_NODELETE BIT IS NORMALLY CLEARED.  I.E. THE ONE BEING
762 ;        REMOVED WAS NOT DELETABLE BUT THE ONE BEING INSERTED IS.
763 ;
764 ;
765 ;        G^SCH$REMOVACB IS AN ALTERNATE ENTRY POINT THAT SIMPLY REMOVES
766 ;        AN ACB FROM AN AST QUEUE.
767 ;
768 ; CALLING SEQUENCE:
769 ;
770 ;        BSB/JSB G^SCH$SWAPACBS
771 ;        BSB/JSB G^SCH$REMOVACB
772 ;        NOTE:  THESE ROUTINES MUST BE CALLED AT IPL$_SYNCH, WITH THE
773 ;                SCHED DATABASE LOCKED.
774 ;
775 ; INPUT PARAMETERS:
776 ;
777 ;        R2      ADDRESS OF ACB TO INSERT (G^SCH$SWAPACBS ONLY)
778 ;        R5      ADDRESS OF ACB TO REMOVE
779 ;
780 ; OUTPUT PARAMETERS:
781 ;
782 ;        NONE
783 ;--
784
785          UNIVERSAL_SYMBOL        SCH$SWAPACBS
786 ;SCH$SWAPACBS::
787          INSQUE  (R2),(R5)               ; INSERT NEW ACB AFTER OLD ONE
788
789          UNIVERSAL_SYMBOL        SCH$REMOVACB
790 ;SCH$REMOVACB::
791          REMQUE  (R5),R5                 ; REMOVE OLD ACB
792          RSB
793
794
795
796          .END
```

2 RSE.LIS

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 0
Table of contents

(2)       40        HISTORY                 ; DETAILED
(3)      223        DECLARATIONS
(4)      431        SCH$RSE - REPORT SYSTEM EVENT
(5)      549        EVENTW - Calculate and apply wait credit
(6)      652        State and event dependent action routines
(7)      716        SCH$UNWAIT - DECREMENT COUNT IN WAIT QUEUE
(8)      775        SITUATIONAL PRIORITY INCREMENT TABLE
(9)      800        SCH$CHSE - CHANGE STATE TO EXECUTABLE
(10)    1008        SWPO - SWAP OUT SIMPLE NON-EXECUTABLE
(11)    1026        SCH$WAIT - PLACE PROCESS IN SELECTED WAIT QUEUE
(12)    1109        SCH$PIXSCAN - Process priority aging
(13)    1262        SCH$ONE_SEC - Perform periodic, 1 second processing
(14)    1380        SCH$QEND - QUANTUM END ROUTINE
(15)    1662        SENDAST - Send AST to process
(17)    1713        SCH$WAKE - WAKE PROCESS INTERNAL
(18)    1752        SCH$SWPWAKE - WAKE SWAPPER PROCESS
(19)    1796        SCH$CHANGE_CUR_PRIORITY  - Modify priority of running process
(20)    1852        SCH$CUR_TO_COM - Make the current process COM
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 1
X-32U5 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (1)

     1          .TITLE  RSE - REPORT SYSTEM EVENT
     2          .IDENT  'X-32U5'
     3
     4 ;
     5 ;*****************************************************************************
     6 ;*                                                                           *
     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
     9 ;*  ALL RIGHTS RESERVED.                                                     *
    10 ;*                                                                           *
    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
    16 ;*  TRANSFERRED.                                                             *
    17 ;*                                                                           *
    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
    20 ;*  CORPORATION.                                                             *
    21 ;*                                                                           *
    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
    24 ;*                                                                           *
    25 ;*                                                                           *
    26 ;*****************************************************************************
    27
    28 ;++
    29 ; FACILITY: EXECUTIVE, SCHEDULER
    30 ;
    31 ; ABSTRACT:
    32 ;       THIS MODULE CONTAINS THE SYSTEM EVENT REPORTING ROUTINES AND
    33 ;       THEIR SUPPORTING SUBROUTINES.
    34 ;
    35 ; ENVIRONMENT:
    36 ;       MODE = KERNEL
    37 ;--
    38 ;
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 2
X-32U5 HISTORY ; DETAILED 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (2)

   40          .SBTTL  HISTORY                 ; DETAILED
   41 ;
   42 ; AUTHOR:       R. HUSTVEDT      CREATION DATE: 6-SEP-76
   43 ;
   44 ; MODIFIED BY:
   45 ;
   46 ;       X-32U5  WWMC00U5        Wayne Cardoza           14-Sep-1988
   47 ;               Support for implicit affinity.
   48 ;
   49 ;       X-32U4  SSA0015         Stan Amway                      1-Sep-1988
   50 ;               Completely tie off use of special scheduling control flags.
   51 ;
   52 ;       X-32U3  WMC0032         Wayne Cardoza                  15-Jul-1988
   53 ;               Use new preemption masks.
   54 ;               New routine to call when changing priority of running process.
   55 ;               New routine to change a process form CUR to COM.
   56 ;
   57 ;       X-32U1  SSA0014         Stan Amway                      1-Jul-1988
   58 ;               In SCH$SWPWAKE, selectively test bits in SCH$GB_SIP. Only
   59 ;               the MPW and SIP bits are relevant to the decision being made.
   60 ;
   61 ;       X-32    SSA0013         Stan Amway                     22-Jan-1988
   62 ;               Change TOCOM_BOOST from 5 to 4. This restores pre-V5 priority
   63 ;               boost for terminal output completion.
   64 ;
   65 ;       X-31    SJF             Stu Farnham                    29-Dec-1987
   66 ;               Consider only ACTIVE CPUs in realtime preemption scan.
   67 ;
   68 ;       X-30    SSA-SJF         Stan Amway-Stu Farnham         23-Nov-1987
   69 ;               In SCH$CHSE(P), use IP preemption logic in the case
   70 ;               where a) candidate process cannot run on current CPU
   71 ;               (either due to affinity or priority), and b) no idle
   72 ;               CPUs are available, but priority and affinity would
   73 ;               allow the process to execute on another CPU.
   74 ;
   75 ;       X-29    SSA0012         Stan Amway              24-Sep-1987
   76 ;               Move definitions of bits in scheduling control
   77 ;               longword to this module.
   78 ;
   79 ;       X-28    SSA0011         Stan Amway               8-Sep-1987
   80 ;               Fix off by 1 bug in selective AWSA code in SCH$QEND.
   81 ;               Check for suitable idle CPUs when making preemption
   82 ;               decisions.
   83 ;
   84 ;       X-27    SSA0010         Stan Amway              28-Jul-1987
   85 ;               Fix EVENTW logic that passes absolute priority boost,
   86 ;               rather than the priority increment class, to SCH$CHSE2.
   87 ;
   88 ;       X-26    SF00026         Stephen Fiorelli        15-Jul-1987
   89 ;               Large working set support
   90 ;
   91 ;       X-25    SSA0009         Stan Amway              23-Mar-1987
   92 ;               Fix bug in EVENTW logic that left high 3 bytes
   93 ;               of PHD address in R0 when calculating new process
   94 ;               priority using byte arithmetic. Caused BBSS R0,...
   95 ;               in SCH$CHSEP to incur access violation.
   96 ;
```

```
;               Make sure PHD is resident before attempting to execute
;               wait credit algorithm, which needs to adjust quantum
;               value in PHD.
;
;       X-24    SSA0008         Stan Amway              17-Mar-1987
;       X-23    SSA0007         Stan Amway              10-Mar-1987
;               Maintain and use PCB$L_PIXHIST.
;
;               In quantum end processing, announce current process priority
;               and selectively force a reschedule if priority demotion
;               enabled at quantum end.
;
;       X-22    SSA0006         Stan Amway               2-Mar-1987
;         21    Use MAXCLASSPRI, MINCLASSPRI, and MINPRPRI to logically
;               partition non-realtime priority range for optional
;               scheduling policies.
;
;       X-20    ROW0594         Ralph O. Weber          11-FEB-1987 17:26
;               Update the reference to SCH$AQ_WQHDR in SCH$UNWAIT to reflect
;               the fact that the SCH$AQ_WQHDR no longer includes the missing
;               zero process state WQH entry computed into its value.  That is
;               subtract WQH$C_LENGTH from SCH$AQ_WQHDR here, because its no
;               longer subtracted in the SYS.STB value of SCH$AQ_WQHDR.
;
;       X-19    ROW0586         Ralph O. Weber          03-FEB-1987 16:08
;               Fix truncation error.
;
;       X-18    RNG5018         Rod Gamache             28-Jan-1987
;               Remove reference to CPU$L_PFAILTIM.
;
;       X-17    SSA0005         Stan Amway               5-Jan-1987
;               Add latent support for queue-dependent quantums and
;               wait credit priority adjustment.
;
;               Merged changes for Rod Gamache and Stu Farnham:
;
;               SJF             Stu Farnham             8-Jan-1987
;               Preserve R1 in affinity loop in CHSE.
;
;               RNG5017         Rod Gamache             7-Jan-1987
;               Fix modulo arithmetic in previous edit.
;
;       X-16    SJF             Stu Farnham             22-Dec-1986
;               Rewrite code which scans for CPU for which process
;               has affinity when blocked due to lack of affinity
;               for current CPU.
;
;       X-15    SSA0004         Stan Amway              12-Dec-1986
;               Optimize SCH$PIXSCAN logic for uniprocessor.
;
;       X-14    JWT0272         Jim Teague              11-Dec-1986
;               Fix a bug in state-matching logic when determining
;               whether or not to deliver ASTs to suspended processes.
;
;       X-13    JWT0264         Jim Teague               8-Dec-1986
;               Part of SOFT susupend implementation.  Don't ignore
;               AST events if the process is in a soft suspend state.
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 4
X-3205 HISTORY ; DETAILED 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (2)

  154 ;                EXEC and KERNEL mode ASTs can be delivered.
  155 ;
  156 ;        X-12    SSA0003         Stan Amway              24-Nov-1986
  157 ;                Honor non-preemptive scheduling control flag.
  158 ;
  159 ;        X-11    SJF             Stu Farnham             31-Oct-1986
  160 ;         -10    Fix spellos.
  161 ;
  162 ;        X-9     RNG0009         Rod Gamache             29-Oct-1986
  163 ;                Change IPL$_SCHED to IPL$_RESCHED.
  164 ;
  165 ;        X-8     SSA0002         Stan Amway              28-Sep-1986
  166 ;                Minor tweaks to PIXSCAN algorithm.
  167 ;
  168 ;                Add usage of the CPU scheduling control flag NO_CHSE_PRIADJ.
  169 ;
  170 ;                Do selective WS reduction at quantum end events, iff,
  171 ;                PFRATL=0, WSDEC <> 0, and current free list size < BORROWLIM.
  172 ;                By allowing WS reduction at quantum end, priority boosting
  173 ;                can work independently of the swapper's dormant process
  174 ;                notion. Potentially this also helps avoid sudden swapper
  175 ;                trimming activity by recognizing memory shortages and
  176 ;                reacting to them sooner.
  177 ;
  178 ;        X-7     SJF             Stu Farnham             22-Oct-1986
  179 ;                Check for affinity in CHSE.
  180 ;
  181 ;        X-6     SSA0001         Stan Amway              22-Sep-1986
  182 ;                Store EXE$GL_ABSTIM_TICS in PCB$L_ONQTIME in SCH$QEND.
  183 ;                This records the time at which the process was placed
  184 ;                on the COM/COMO queue, ignoring any priority boosting.
  185 ;
  186 ;                Expand priority increment class table.
  187 ;
  188 ;                Make priority comparison strictly > for pre-emption.
  189 ;
  190 ;                Use scheduling control flag settings to parameterize
  191 ;                the scheduling policy.
  192 ;
  193 ;                Move routine SCH$WAIT from SYSWAIT.MAR to this module.
  194 ;                Store EXE$GL_ABSTIM_TICS instead of EXE$GL_ABSTIM
  195 ;                in PCB$L_WAITIME.
  196 ;                Remove ASMP conditional assembly code (per Rod Gamache).
  197 ;
  198 ;                Move PIXSCAN code from TIMESCHDL.MAR to this module.
  199 ;                Rewrite algorithm to work with SMP and to be more
  200 ;                responsive.
  201 ;
  202 ;        X-5     WMC0002         Wayne Cardoza           15-Aug-1986
  203 ;                Change some JSBs to BSBWs.
  204 ;
  205 ;        X-1D1   SF04001         Stephen Fiorelli        10-Dec-1985
  206 ;                Resolve conflicts from initial merge of exec reorg
  207 ;                thread and mainline (4.4 BL7).
  208 ;
  209 ;        X-2     WMC0001         Wayne Cardoza           24-Sep-1985
  210 ;                Don't allow working set decrement if PHD$V_NO_WS_CHNG is set.
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 5
X-32U5 HISTORY ; DETAILED 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (2)

  211 ;
  212 ;       V04-002 TCM0003         Trudy C. Matthews       23-Aug-1985
  213 ;               Make RSE's AES1 and AES2 psect attributes correspond to
  214 ;               those of the EXEC$NONPAGED_DATA psect, so as to not
  215 ;               create a separate image section for these psects.
  216 ;
  217 ;       V04-001 TCM0002         Trudy C. Matthews       30-Apr-1985
  218 ;               Define EVT$_ event codes in an SDL file, not in EVENT
  219 ;               macro.
  220 ;
  221
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 6
X-3205 DECLARATIONS 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (3)

  223          .SBTTL  DECLARATIONS
  224
  225 ;
  226 ; INCLUDE FILES:
  227 ;
  228          $ACBDEF                         ; DEFINE AST CONTROL BLOCK
  229          $CEBDEF                         ; DEFINE COMMON EVENT BLOCK
  230          $CPBDEF                         ; CAPABILITIES
  231          $CPUDEF                         ; DEFINE PER-CPU DATA STRUCTURE OFFSETS
  232          $DYNDEF                         ; DEFINE STRUCTURE TYPE CODES
  233          $EVTDEF                         ; DEFINE EVENT CODES
  234          $IPLDEF                         ; IPL DEFINITIONS
  235          $PCBDEF                         ; PCB DEFINITIONS
  236          $PHDDEF                         ; PROCESS HEADER DEFINITIONS
  237          $PRDEF                          ; PROCESSOR REGISTER DEFS
  238          $PRIDEF                         ; PRIORITY INCREMENT CLASSES
  239          $PSLDEF                         ; Processor status longword fields
  240          $SPLCODDEF                      ; Various SMP definitions
  241          $SSDEF                          ; DEFINE STATUS CODES
  242          $STATEDEF                       ; STATE DEFINITIONS
  243          $WQHDEF                         ; WAIT QUEUE HEADER DEFINITIONS
  244 ;
  245 ; MACROS:
  246 ;
  247          .MACRO  EVENT,EVTN,STATLIST,EACTION,CONT=0
  248
  249 .=STACT+<2*EVT$_'EVTN>
  250          .WORD   EACTION-STACT
  251          .PSECT  AES2,BYTE,NOEXE,PIC
  252 STMSK=CONT
  253          .IRP    ST,<STATLIST>
  254 STMSK=STMSK+<1@SCH$C_'ST>
  255          .ENDR
  256 MASK_'EVTN:
  257          .LONG   STMSK
  258          .PSECT  AES1,BYTE,NOEXE,PIC
  259 .=STACT+<2*EVT$_MAXEVT>
  260          .ENDM   EVENT
  261
  262 ;
  263 ;        GENERATE MASK FOR WAIT STATES
  264 ;
  265 ;        GMASK   STATENAME
  266 ;
  267
  268          .MACRO  GMASK,STATE
  269 ST=SCH$C_'STATE
  270 WAITST=WAITST+<1@ST>
  271          .ENDM   GMASK
  272
  273 ;
  274 ;        Macro to set correct priority increment for
  275 ;        a given priority increment class
  276 ;
  277          .MACRO  PRIORITY_BOOST,PINC,VALUE
  278 DOT=.
  279          .=B_PINC+PRI$_'PINC
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 7
X-32U5 DECLARATIONS 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (3)

          .BYTE   VALUE
.=DOT
          .ENDM

;
; EQUATED SYMBOLS:
;
WAITST=0                                ; INITIALIZE WAIT STATE MASK
ASTEXIT=0                               ; AST EXIT CHANGE MODE CODE

TICOM_BOOST=6                           ; Boost for terminal input  completion
MAX_BOOST=6                             ; NB: must be equal to TICOM_BOOST
NO_BOOST=5                              ; Priority at or above which no boost is giv
TOCOM_BOOST=4                           ; Boost for terminal output completion

;
; OWN STORAGE:
;

          DECLARE_PSECT   EXEC$NONPAGED_DATA
;
; Define the control bits in SCH$GL_CTLFLAGS and its local copy, SCH$$GL_CTLFLAGS
;
          $GBLINI GLOBAL
;
; NB: Code relies on the following properties of the bits defined here.
;
;         NO_RESCHED_PRIADJ must be bit 0 (SCHED.MAR)
;         WAIT_CREDIT must the next higher number bit than PRI_QUANT (RSE.MAR)
;
          $VIELD  SCH,0,<-
                  <NO_RESCHED_PRIADJ,,M>,-; Don't decrease priority on reschedule
                  <QEND_PRIADJ,,M>,-      ; Decrease priority at quantum end
                  <PRI_QUANT,,M>,-        ; Use priority-dependent quantum values
                  <WAIT_CREDIT,,M>,-      ; Manipulate priorities via wait credit, no
                  <NO_PIXSCAN,,M>,-       ; Don't use PIXSCAN mechanism for process ag
                  <NO_CHSE_PRIADJ,,M>,-   ; Don't boost process priorities in event re
                  <PIXSCAN_LOWPRI,,M>,-   ; Lowest priority process to rcv proc. aging
                  <NO_QEND_PREEMPT,,M>,-  ; Don't preemptively reschedule at quantum e
                  <PREEMPT_RESUME,,M>,-   ; Requeue PCB at head of queue when process
          >

;
; The following cells are updated every second by the SCH$ONE_SEC routine. In
; this way, the relevant SYSGEN parameters can remain dynamic, while permitting
; time-critical code in this module to use W^ addressing modes and precomputed
; table values. Since the SYSGEN parameters are changed very infrequently on
; the running system, the table values are recalculated only if the basic
; quantum value (SCH$GW_QUAN) changes.
;
SCH$$GL_CTLFLAGS::
          .LONG   0                     ; Local copy of dynamic SYSGEN
                                        ; parameter

;
; Minimum wait credit to consider. Set to MAX(IOTA, MINWC).
;
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 8
X-32U5 DECLARATIONS 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (3)

  337         MINWC=10                        ; 100 ms
  338 SCH$GL_MINWC::
  339         .LONG   0
  340
  341 ;****************************************************************************
  342 ; The following cells and tables are initialized to contain values that are a
  343 ; function of a) the base quantum value (SCH$GW_QUAN), b) the scheduling
  344 ; control flag (SCH$V_PRI_QUANT) that determines whether a single quantum, or
  345 ; queue dependent quantums are desired, and c) the scheduling control flag
  346 ; (SCH$V_WAIT_CREDIT) that determines whether wait credit is used to manipulate
  347 ; process priority. Algorithmically, scheduling code treats all cases uniformly
  348 ; using the stored table values as decision variables.
  349 ;
  350 ; NB: The wait algorithms assume that the contents of any table cell and
  351 ;     SCH$GL_MAXWC is always less than 2**15. This allows the code to do
  352 ;     SIGNED WORD arithmetic when dealing with quantum and wait credit values.
  353 ;****************************************************************************
  354
  355                                 ; Max quantum, no options
  356         MAX_QUANT     = 32767
  357                                 ; Max quantum, either option selected
  358         MAX_QUANT_OPT = <MAX_QUANT/<MAX_BOOST+1>>
  359
  360 ;
  361 ; The following table and the code that accesses it assume that the control
  362 ; flag SCH$V_PRI_QUANT immediately precedes SCH$V_WAIT_CREDIT.
  363 ;
  364 SCH$GW_MAX_QUANT::
  365                                 ; WAIT_CREDIT   PRI_QUANT
  366         .WORD   MAX_QUANT       ;       0           0
  367         .WORD   MAX_QUANT_OPT   ;       0           1
  368         .WORD   MAX_QUANT_OPT   ;       1           0
  369         .WORD   MAX_QUANT_OPT   ;       1           1
  370 ;
  371 ; Queue-dependent, basic quantum multiplier table
  372 ;
  373 SCH$GL_QUANT_MULT::
  374         .LONG   32              ; Base priority
  375         .LONG   16              ; Base priority + 1
  376         .LONG    8              ; Base priority + 2
  377         .LONG    4              ; Base priority + 3
  378         .LONG    2              ; Base priority + 4
  379         .LONG    1              ; Base priority + 5
  380 ; The last entry must have a multiplier of 1
  381         .LONG    1              ; Base priority + MAX_BOOST
  382
  383         ASSUME  <<.-SCH$GL_QUANT_MULT>/4> EQ <MAX_BOOST+1>
  384
  385 ;
  386 ; Maximum wait credit to consider. Set to SCH$GW_MAXWC[0] + SCH$GW_QUANTUM[0].
  387 ;
  388 SCH$GL_MAXWC::
  389         .LONG   0
  390
  391 ;
  392 ; The next table contains the maximum wait credit that can be applied
  393 ; at each relative priority. It is used as a quick check to determine
```

```
394 ; if the longer wait credit algorithm can be completely avoided.
395 ;
396 SCH$GW_MAXWC::
397         .WORD   0[MAX_BOOST+1]
398 ;
399 ; The next table contains the quantum reset value for each relative
400 ; priority.
401 ;
402 SCH$GW_QUANTUM::
403         .WORD   20[MAX_BOOST+1]         ; Preset to 200 ms quantum
404 SCH$$GW_QUAN==.-2
405
406 ;
407 ; NB: The following 3 cells must be internal priority format.
408 ;
409 SCH$$GB_MINCLASSPRI::
410         .BYTE   <31-1>                  ; Local copy of the
411                                         ; SYSGEN parameter MINCLASSPRI
412
413 SCH$$GB_MAXCLASSPRI::
414         .BYTE   <31-7>                  ; Local copy of the
415                                         ; SYSGEN parameter MAXCLASSPRI
416
417 SCH$$GB_MINPRPRI::
418         .BYTE   <31-7>                  ; Local copy of the
419                                         ; SYSGEN parameter MINPRPRI
420
421         .ALIGN  LONG
422 SCH$GB_RT_CPUID::                       ; Cells used by IP preemption logic
423         .BYTE   0
424 SCH$GB_RT_CURPRI::
425         .BYTE   0
426         .PSECT  AES2,BYTE,NOEXE,PIC     ; STATE EVENT MASK PSECT
427 STET=   .                               ; BASE OF STATE EVENT TABLE
428
429         .PSECT  AES1,BYTE,NOEXE,PIC
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 10
X-32U5 SCH$RSE - REPORT SYSTEM EVENT 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (4)

  431          .SBTTL  SCH$RSE - REPORT SYSTEM EVENT
  432
  433 ;++
  434 ; FUNCTIONAL DESCRIPTION:
  435 ;       SCH$RSE RECEIVES SYSTEM EVENT REPORTS FROM VARIOUS SOURCES
  436 ;       AND PERFORMS THE APPROPRIATE ACTION FOR THE SPECIFIED PROCESS.
  437 ;       EVENT REPORTING MUST BE PERFORMED WITH IPL=IPL$_SYNCH.
  438 ;       AS A SIDE EFFECT OF AN EVENT REPORT, THE RESCHEDULING INTERRUPT
  439 ;       MAY BE TRIGGERED IF APPROPRIATE.
  440 ;
  441 ; CALLING SEQUENCE:
  442 ;       BSB/JSB G^SCH$RSE
  443 ;       .BYTE   EVT$_EVENTNAME
  444 ;
  445 ;       THIS CALLING SEQUENCE IS GENERATED BY THE RPTEVT SYSTEM MACRO
  446 ;
  447 ;       REPEVT  EVENTNAME
  448 ;
  449 ; INPUT PARAMETERS:
  450 ;       R2 - SITUATIONAL PRIORITY INCREMENT CLASS NUMBER
  451 ;       R4 - PCB ADDRESS OF PROCESS FOR WHICH EVENT IS REPORTED
  452 ;
  453 ;       EVENT NUMBER CONTAINED IN BYTE LOCATED BY ADDRESS AT TOP
  454 ;       OF STACK.  @(SP)
  455 ;
  456 ; IMPLICIT INPUTS:
  457 ;       NONE
  458 ;
  459 ; OUTPUT PARAMETERS:
  460 ;       NONE
  461 ;
  462 ; IMPLICIT OUTPUTS:
  463 ;       NONE
  464 ;
  465 ; COMPLETION CODES:
  466 ;       NONE
  467 ;
  468 ; SIDE EFFECTS:
  469 ;       A RESECHEDULING INTERRUPT MAY BE REQUESTED IF THE SPECIFIED
  470 ;       PROCESS IS HIGHER IN PRIORITY THAN THE CURRENT PROCESS.
  471 ;
  472 ; ENVIRONMENT:
  473 ;       IPL = IPL$_SYNCH, SCHED DATABASE LOCKED
  474 ;--
  475
  476          UNIVERSAL_SYMBOL        SCH$RSE
  477 ;SCH$RSE::                                ; REPORT SYSTEM EVENT
  478          MOVZBL  @(SP),R3                ; GET EVENT NUMBER
  479          INCL    (SP)                    ; UPDATE RETURN ADDRESS
  480          MOVZWL  PCB$W_STATE(R4),R1      ; GET CURRENT STATE NUMBER
  481 10$:     MOVL    L^STET[R3],R0           ; GET STATE MASK FOR EVENT
  482          BBS     R1,R0,ACTION            ; DO ACTION IF STATE BIT SET
  483          INCL    R3                      ; CHECK NEXT ACTION
  484          BLBS    R0,10$                  ; IF CONTINUATION
  485          RSB                             ; OTHERWISE IGNORE EVENT
  486
  487 ACTION: CASEL   R3,#0,S^#EVT$_MAXEVT-1  ; SWITCH ON EVENT NUMBER(UPDATED)
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 11
X-32U5 SCH$RSE - REPORT SYSTEM EVENT 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (4)

  488 STACT:                                    ; BASE OF ACTION TABLE
  489         EVENT   AST,<-                    ; AST EVENT
  490                 CEF,-                     ; COMMON EVENT FLAG WAIT
  491                 COLPG,-                   ; COLLIDED PAGE WAIT
  492                 FPG,-                     ; FREE PAGE WAIT
  493                 HIB,-                     ; RESIDENT HIBERNATE
  494                 HIBO,-                    ; NON-RESIDENT HIBERNATE
  495                 LEF,-                     ; LOCAL EVENT FLAG WAIT
  496                 LEFO,-                    ; LOCAL EVENT FLAG WAIT (NON-RES)
  497                 MWAIT,-                   ; MUTEX WAIT
  498                 PFW,-                     ; PAGE FAULT WAIT
  499                 SUSP,-
  500                 SUSPO,-
  501                 >,AST_EVENT               ; AST EXECUTABLE STATE CHANGE
  502 EVT$_COLPGA==EVT$_AST                     ; USE SAME EVENT FOR COLLIDED PAGE AVAIL
  503
  504
  505         EVENT   EVENT,<LEF>,LEFEVT,CONT=1       ; EVENT FLAG SETTING
  506         EVENT   CEF,<CEF>,CEFEVT,CONT=1 ; COMMON EVENT FLAG SET
  507         EVENT   LEFO,<LEFO>,EVENTE              ; NON-RESIDENT LOCAL EVENT
  508
  509         EVENT   FPGA,<-                   ; FREE PAGE AVAILABLE
  510                 FPG,-                     ; FREE PAGE WAIT STATE
  511                 >,EVENTF                  ; EXECUTABLE STATE CHANGE
  512         EVENT   WAKE,<-                   ; WAKE EVENT
  513                 HIB,-                     ; RESIDENT HIBERNATE
  514                 HIBO,-                    ; NON-RESIDENT HIBERNATE
  515                 >,EVENTE                  ;
  516
  517
  518         EVENT   RESUME,<-                 ; RESUME EVENT
  519                 SUSP,-                    ; RESIDENT SUSPENDED
  520                 SUSPO,-                   ; NON-RESIDENT SUSPENDED
  521                 >,EVENTE                  ; EXECUTABLE
  522
  523
  524         EVENT   PFCOM,<-                  ; PAGE FAULT COMPLETE EVENT
  525                 PFW,-                     ; PAGE FAULT WAIT
  526                 >,EVENTE                  ; EXECUTABLE
  527
  528         EVENT   SETPRI,<-                 ; SET PRIORITY EVENT
  529                 COM,-                     ; RESIDENT COMPUTE
  530                 COMO,-                    ; NON-RESIDENT COMPUTE
  531                 >,EVENTF                  ; EXECUTABLE
  532
  533         EVENT   SWPOUT,<-                 ; SWAP OUT EVENT
  534                 HIB,-                     ; RESIDENT HIBERNATE
  535                 LEF,-                     ; RESIDENT LOCAL EVENT FLAG WAIT
  536                 SUSP,-                    ; RESIDENT SUSPENDED
  537                 >,SWPO,CONT=1             ; SIMPLE SWAP OUT
  538         EVENT   SWPOUTE,<-                      ; SWAP OUT EVENT CONINUATION
  539                 COM,-                     ; RESIDENT COMPUTE
  540                 >,SWPOE                   ; EXECUTABLE OUTSWAP
  541
  542 ;
  543 ;       IF CASE FALLS THROUGH, THEN BUGCHECK WITH ILLEGAL EVENT
  544 ;       NUMBER.
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 12
X-32U5 SCH$RSE - REPORT SYSTEM EVENT 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (4)

  545 ;
  546          BUG_CHECK ILLEVTNUM,FATAL       ; ILLEGAL EVENT NUMBER
  547
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 13
X-32U5 EVENTW - Calculate and apply wait credit 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RS

          .SBTTL  EVENTW - Calculate and apply wait credit
;
;       The logic is optimimized assuming that very short waits (primarily
;       disk I/O) and very long waits, in that order, are the most frequent.
;       The former case requires no boost, and the latter, a maximum boost.
;       Beyond these cases, an additional assumption is that wait credit
;       generally will not be enough to cause queue promotion.
;
;       NB: The PHD is assumed to be resident upon entry to this code.
;
EVENTW:
          SUBL3   PCB$L_WAITIME(R4),-     ; Compute time that process was waiting
                  G^EXE$GL_ABSTIM_TICS,R0
          ADDL2   R0,PCB$L_ONQTIME(R4)    ; Remove wait time from on queue time

          CMPL    R0,W^SCH$GL_MINWC       ; Enough wait credit to even consider ?
          BGEQU   10$                     ; BR if enough

200$:     CLRL    R2                      ; Set no priority boost
250$:     PUSHAB  W^SCH$CHSE2             ; Use alternate entry to force boost
          BRW     SCH$UNWAIT

300$:     MOVL    PCB$L_PHD(R4),R2        ; Address PHD
          MNEGW   W^SCH$GW_QUANTUM+<MAX_BOOST*2>,-
                  PHD$W_QUANT(R2)         ; Set quantum to maximum for queue
;
; Note: If the process has a base priority > 9, than the priority boosting
; logic will set the current priority to the process' base priority. No attempt
; is made here to avoid that ANOMALY, because a) it is logically consistent
; with the pre-V5 priority boosting only scheme, b) it is not critical that
; such processes receive the boost, and c) it is a special case that we wish
; to avoid for algorithmic simplicity.
;
          ASSUME  MAX_BOOST EQ TICOM_BOOST
          MOVL    #PRI$_TICOM,R2          ; Set maximum priority boost
          BRB     250$

10$:      CMPL    R0,W^SCH$GL_MAXWC       ; Wait credit >= extent of table ?
          BGEQU   300$                    ; BR if yes; give maximum boost
          SUBB3   PCB$B_PRI(R4),-         ; Calculate relative priority to base
                  PCB$B_PRIB(R4),R1
          BLSS    200$                    ; BR if current priority < base
          CMPB    #NO_BOOST,R1            ; Is priority too high for a boost ?
          BLEQ    200$                    ; BR if yes
          CMPB    #PRI$_TICOM,R2          ; Is this a terminal input completion ?
          BEQL    80$                     ; BR if yes
25$:
          MOVL    PCB$L_PHD(R4),R2        ; Get address of process header
          SUBW2   PHD$W_QUANT(R2),R0      ; Adjust remaining quantum by wait credit
          MOVZBL  R1,R1                   ; Get a LW copy for indexing
          CMPW    R0,W^SCH$GW_QUANTUM[R1] ; Result >= queue quantum ?
          BGEQ    400$                    ; BR if yes
          MNEGW   R0,PHD$W_QUANT(R2)      ; Else store adjusted quantum
          MOVZBL  PCB$B_PRI(R4),R2        ; Retain current priority

50$:      PUSHAB  B^100$                  ; Make SCH$UNWAIT exit via linkage code
          BRW     SCH$UNWAIT
```

```
606
607 80$:    CMPL    #100,R0                 ; Wait credit >= 1 second ?
608         BGTR    25$                     ; BR if not
609         BRB     300$                    ; Else give maximum boost
610
611 100$:   MOVL    R2,R0                   ; Restore precomputed priority
612         BRW     SCH$CHSEP               ; and change process state to COM(O)
613
614 ;
615 ; R0 = remaining quantum (positive), adjusted by wait credit
616 ; R1 = process priority relative to base (range 0-4)
617 ; R2 = PHD address
618 ;
619 400$:
620         SUBW2   W^SCH$GW_QUANTUM[R1],R0 ; Compute excess wait credit
621         INCL    R1                      ; Boost priority by 1
622 ;
623 ; The following test also detects the case where the PHD quantum is greater
624 ; than the maximum allowed for the relative queue. This can only result
625 ; from a) current priority adjustments, caused by PIXSCAN processing, or
626 ; system event priority boosting (unlikely, given wait crediting is enabled);
627 ; or b) base priority adjustments caused by explicit user request
628 ; ($SET PROCESS/PRIORITY), program request ($SETPRI), or class scheduling.
629 ;
630         CMPW    R0,W^SCH$GW_MAXWC[R1]   ; Remaining credit >= maximum for queue
631         BGEQ    300$                    ; Yes, give maximum boost
632 410$:   CMPW    R0,W^SCH$GW_QUANTUM[R1] ; Remaining credit <= queue quantum
633         BLEQ    450$                    ; BR if yes
634         SUBW2   W^SCH$GW_QUANTUM[R1],R0 ; Compute excess wait credit
635         INCL    R1                      ; Boost priority by 1
636         BRB     410$
637
638 450$:   CMPB    #NO_BOOST,R1            ; Is resultant priority = threshold ?
639         BEQL    470$                    ; BR if yes
640         MNEGW   R0,PHD$W_QUANT(R2)      ; Reset quantum value
641         BGEQ    480$                    ; BR if remaining quantum too small
642 460$:   CLRL    R2                      ; Pre-zero resultant priority
643         SUBB3   R1,PCB$B_PRIB(R4),R2    ; Calculate new priority
644         BRB     50$                     ; Join common exit code
645
646 470$:   DECL    R1                      ; Force process to next lower priority
647 480$:   MNEGW   W^SCH$GW_QUANTUM[R1],-  ; Use full queue quantum
648                 PHD$W_QUANT(R2)
649         BRB     460$
650
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 15
X-32U5 State and event dependent action routine 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RS

652          .SBTTL  State and event dependent action routines
653 ;
654 ;        COMMON EVENT FLAG SET
655 ;
656 CEFEVT:                                      ;
657          BLBC    PCB$L_STS(R4),EVENTE       ; BR IF NOT RESIDENT
658 LEFEVT:                                      ; LOCAL EVENT FLAG SET FOR RESIDENT PROCESS
659          MOVL    PCB$L_PHD(R4),R0           ; POINT TO PHD
660          ADDL    #4,PHD$L_PC(R0)            ; SKIP PC OVER CHMK INSTRUCTION
661          MOVZWL  #SS$_NORMAL,PHD$L_R0(R0)   ; SET NORMAL COMPLETION FOR WAIT
662          BRB     EVENTE
663 ;
664 ;  AST Event and a suspended process...is it in a
665 ;        soft suspend state?  If so, report the event;
666 ;        otherwise ignore it
667
668 SOFT:    BBS     #PCB$V_SOFTSUSP,PCB$L_STS(R4),EVENTE
669          RSB
670
671 ;  AST event...is this a suspended process?  If not, report
672 ;        the event by falling through to EVENTE...otherwise
673 ;        see if this is a soft suspend.
674 ;
675 ;  How does this work?  Note that every event above (i.e., AST,
676 ;        RESUME, SWPOUT, etc.) now has a label on its state mask.
677 ;        (Look in the macro EVENT.)  The mask for RESUME has a bit
678 ;        set for SUSP and SUSPO, and that's exactly what we want to
679 ;        check for: if the state in R1 corresponds to either of those
680 ;        bits being set, then check the SOFTSUSP bit before deciding
681 ;        whether or not to deliver the AST.
682 ;
683 AST_EVENT:
684          BBS     R1,W^MASK_RESUME,SOFT      ; If SUSP or SUSPO, check SOFTSUSP BIT
685 ;
686 ;        EVENT EXECUTABLE ACTION ROUTINE
687 ;
688
689 EVENTF:                                     ; ACTIVATE WITH NO WAIT TIME ACCOUNTING
690          BBC     #4,PCB$B_PRIB(R4),EVENTC;  BR if real-time process
691          BBC     S^#SCH$V_WAIT_CREDIT,-     ; BR if wait credit processing disabled
692                  W^SCH$$GL_CTLFLAGS,EVENTC
693          CLRL    R2                         ; Make sure no priority boost is given
694          BRB     EVENTC                     ; Join common code
695
696 CHECK_EVENTW:
697          CMPB    #EVT$_PFCOM,R3             ; Was event a page fault completion ?
698          BEQL    EVENTC                     ; BR if yes; no wait credit processing
699          BBC     #PCB$V_PHDRES,-
700                  PCB$L_STS(R4),EVENTC       ; BR if PHD not resident
701          BRW     EVENTW
702
703 EVENTE:                                     ; EVENT EXECUTABLE STATE CHANGE
704          BBC     #4,PCB$B_PRIB(R4),EVENTC;  BR if real-time process
705          BBS     S^#SCH$V_WAIT_CREDIT,-     ; BR if wait credit processing enabled
706                  W^SCH$$GL_CTLFLAGS,CHECK_EVENTW
707 EVENTC:
708          SUBL3   PCB$L_WAITIME(R4),-        ; Compute time that process was waiting
```

```
                        G^EXE$GL_ABSTIM_TICS,R0
                ADDL2   R0,PCB$L_ONQTIME(R4)    ; Remove wait time from on queue time
;
                PUSHAB  B^SCH$CHSE              ; MAKE UNWAIT EXIT THROUGH CHSE
;               BRB     SCH$UNWAIT              ; AND FALL INTO UNWAIT
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 17
X-32U5 SCH$UNWAIT - DECREMENT COUNT IN WAIT QUE 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RS

          .SBTTL  SCH$UNWAIT - DECREMENT COUNT IN WAIT QUEUE
;++
;
; FUNCTIONAL DESCRIPTION:
;       SCH$UNWAIT DECREMENTS THE NUMBER OF PROCESSES IN THE WAIT
;       QUEUE SELECTED BY THE SPECIFIED PCB AND STATE VALUE.
;
; CALLING SEQUENCE:
;       BSB/JSB G^SCH$UNWAIT
;
; INPUT PARAMETERS:
;       R2 - UNUSED       (PRESERVED)
;       R4 - PCB ADDRESS  (PRESERVED)
;
; IMPLICIT INPUTS:
;       PCB LOCATED BY ADDRESS IN R4
;
; OUTPUT PARAMETERS:
;       R0 - Address of wait queue header (iff process is in a wait state)
;
; IMPLICIT OUTPUTS:
;       COUNT IN WAIT QUEUE HEADER IS DECREMENTED IF STATE IS A WAIT
;       STATE.
;
; ENVIRONMENT:
;       IPL = IPL$_SYNCH, SCHED DATABASE LOCKED
;--

          UNIVERSAL_SYMBOL        SCH$UNWAIT
;SCH$UNWAIT::                              ; DECREMENT PROPER WAIT COUNT
          MOVZWL  PCB$W_STATE(R4),R1        ; Get current process state number
          BBC     R1,B^WAITMSK,20$          ; SKIP OUT IF NOT WAIT STATE
          CMPW    #SCH$C_CEF,R1             ; CHECK FOR COMMON EVENT FLAG WAIT
          BEQL    30$                       ; CEF WAIT
          MULL    #WQH$C_LENGTH,R1          ; COMPUTE BYTE INDEX TO WQ HDR
          MOVAB   G^SCH$AQ_WQHDR -          ; COMPUTE ADDRESS OF WAIT Q HEADER
                   -WQH$C_LENGTH[R1],R0     ;  (note state indicies start at 1)
10$:      DECW    WQH$W_WQCNT(R0)           ; DECREMENT WAIT QUEUE COUNT
20$:      RSB                               ; RETURN

30$:      MOVZBL  PCB$B_WEFC(R4),R0         ; WAIT CLUSTER NUMBER
          MOVL    PCB$L_EFCS(R4)[R0],R0     ; GET CLUSTER ADDRESS
          ADDL    #CEB$L_WQFL,R0            ; POINT TO WAIT QUEUE HEADER
          BRB     10$                       ; GO DECREMENT WAIT COUNT

          GMASK   CEF                       ; COMMON EVENT FLAG
          GMASK   LEF                       ; LOCAL EVENT FLAG WAIT
          GMASK   LEFO                      ; LOCAL EVENT FLAG WAIT
          GMASK   HIB                       ; HIBERNAT WAIT
          GMASK   HIBO                      ; HIBERNATE WAIT
          GMASK   FPG                       ; FREE PAGE WAIT
          GMASK   COLPG                     ; COLLISION PAGE WAIT
          GMASK   PFW                       ; PAGE FAULT WAIT
          GMASK   SUSP                      ; SUSPENDED WAIT
          GMASK   SUSPO                     ; SUSPENDED WAIT
          GMASK   MWAIT                     ; MUTEX WAIT
WAITMSK:.LONG   WAITST                      ; MASK OF WAIT STATES
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 18
X-32U5 SCH$UNWAIT - DECREMENT COUNT IN WAIT QUE 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RS

  773
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 19
X-32U5 SITUATIONAL PRIORITY INCREMENT TABLE 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MP

          .SBTTL  SITUATIONAL PRIORITY INCREMENT TABLE
;
; FIXED DATA:
;               SITUATIONAL PRIORITY INCREMENT TABLE
;                   (INDEXED BY PRIORITY INCREMENT CLASS)
;

PINC_SPARES=5                           ; Spare entries for expansion of table

B_PINC:
                                        ; Init all entries for no priority boost
          .BYTE   0[PRI$_PINC_CNT+PINC_SPARES]

          PRIORITY_BOOST  NULL,0        ; No priority boost
          PRIORITY_BOOST  IOCOM,2       ; I/O completion
          PRIORITY_BOOST  RESAVL,3      ; Resource available
          PRIORITY_BOOST  TOCOM,TOCOM_BOOST ; Terminal output completion
          PRIORITY_BOOST  TICOM,TICOM_BOOST ; Terminal input completion
          PRIORITY_BOOST  TIMER,3       ; Timer request completion


EXESTATE:                               ; EXECUTABLE STATE MASK
          .LONG   <1@SCH$C_COM>!<1@SCH$C_COMO>

```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 20
X-32U5 SCH$CHSE - CHANGE STATE TO EXECUTABLE 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.M

  800          .SBTTL  SCH$CHSE - CHANGE STATE TO EXECUTABLE
  801 ;++
  802 ; FUNCTIONAL DESCRIPTION:
  803 ;       SCH$CHSE CHANGES THE STATE OF A PROCESS, AS REPRESENTED BY
  804 ;       ITS PCB, TO AN EXECUTABLE STATE.  THE RESCHEDULING INTERRUPT
  805 ;       WILL BE TRIGGERED IF THE PROCESS IS RESIDENT AND HAS A PRIOITY
  806 ;       GREATER THAN THAT OF THE CURRENTLY EXECUTING PROCESS.  A
  807 ;       PRIORITY INCREMENT CLASS NUMBER SUPPLIED AS A REGISTER CONTAINED
  808 ;       ARGUMENT IS USED TO COMPUTE THE NEW PROCESS PRIORITY FROM ITS
  809 ;       BASE PRIORITY.
  810 ;
  811 ; CALLING SEQUENCE:
  812 ;       BSB/JSB G^SCH$CHSE
  813 ;
  814 ; INPUT PARAMETERS:
  815 ;       R0 - NEW PRIORITY       (SCH$CHSEP ONLY)
  816 ;       R2 - PRIORITY INCREMENT CLASS NUMBER   (SCH$CHSE ONLY)
  817 ;       R4 - PCB ADDRESS
  818 ;
  819 ; IMPLICIT INPUTS:
  820 ;       SCH$AQ_COMT - COMPUTE QUEUE HEADERS FOR COM,COMO STATES
  821 ;       SCH$GB_PRI  - CURRENT PROCESS PRIORITY.
  822 ;       CPU$B_CUR_PRI  - CURRENT PROCESS PRIORITY IN PER-CPU DATA
  823 ;
  824 ;
  825 ; OUTPUT PARAMETERS:
  826 ;       R2 - ADDRESS OF PER-CPU DATA BLOCK
  827 ;       R3 - R3 (PRESERVED)
  828 ;
  829 ; IMPLICIT OUTPUTS:
  830 ;       SCH$AQ_COMH  -  VECTOR OF COMPUTE QUEUE HEADERS.
  831 ;       SCH$GL_COMQS -  COMPUTE QUEUE SUMMARY BIT VECTOR.
  832 ;
  833 ; COMPLETION CODES:
  834 ;       NONE
  835 ;
  836 ; SIDE EFFECTS:
  837 ;       THE PCB SPECIFIED IS REMOVED FROM ITS PRESENT STATE QUEUE
  838 ;       AND INSERTED IN THE APPROPRIATE COMPUTE QUEUE, COM OR COMO,
  839 ;       AT THE PRIORITY COMPUTED FOR THE SPECIFIED SITUATION CLASS.
  840 ;       THE SUMMARY BIT FOR THE DESTINATION STATE QUEUE IS SET TO
  841 ;       NOTE THAT IT IS OCCUPIED.
  842 ;       IF THE NEW PRIORITY FOR THE PROCESS IS GREATER THAN THAT OF
  843 ;       CURRENT PROCESS AND IT IS RESIDENT, THE RESCHEDULING INTERRUPT
  844 ;       WILL BE TRIGGERED.
  845 ;
  846 ; ENVIRONMENT:
  847 ;       IPL = IPL$_SYNCH, SCHED DATABASE LOCKED
  848 ;--
  849          .ENABLE LSB
  850
  851 ;
  852 ; NB: The BBS that passed control here CANNOT be changed to branch directly
  853 ;     to the instruction 'MOVB PCB$B_PRI(R4),R0'. While at first glance this
  854 ;     would seem correct, it does not correctly handle the case of the current
  855 ;     priority less than the base priority. Other scheduling-related code,
  856 ;     notably the $SETPRI logic, wishes to avoid some hard to handle special
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 21
X-32U5 SCH$CHSE - CHANGE STATE TO EXECUTABLE 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.M

  857 ;     cases and depends on SCH$CHSE to behave this way.
  858 ;
  859 5$:     CLRL    R2                          ; Force NULL priority increment class
  860         BRB     SCH$CHSE2
  861
  862 BADIPL: BUG_CHECK BADRSEIPL,FATAL           ; BAD IPL AT ENTRANCE TO RSE
  863
  864         UNIVERSAL_SYMBOL        SCH$CHSE
  865 ;SCH$CHSE::                                 ;CHANGE TO EXECUTABLE STATE
  866         BBS     S^#SCH$V_NO_CHSE_PRIADJ,-; BR if event reporting priority
  867                 W^SCH$$GL_CTLFLAGS,5$   ; adjustment is disabled
  868 SCH$CHSE2::
  869         CLRL    R0                          ; CLEAR HIGH SUM BITS FOR ADDB
  870         SUBB3   B_PINC[R2],PCB$B_PRIB(R4),R0    ; ADD PRIORITY INCR
  871         CMPB    R0,PCB$B_PRI(R4)            ; CHECK FOR > CURRENT PRI
  872         BLEQ    10$                         ; NO
  873         MOVB    PCB$B_PRI(R4),R0            ; KEEP CURRENT PRIORITY INSTEAD
  874 10$:    CMPB    R0,#16                      ; CHECK FOR RESULT >15
  875         BGEQ    SCH$CHSEP                   ; YES, USE COMPUTED VALUE
  876         MOVB    PCB$B_PRIB(R4),R0           ; KEEP AT BASE IF LESS
  877
  878         .DISABLE LSB
  879
  880 ;
  881 ;       SCH$CHSEP - SUB-ENTRY POINT WITH PRIORITY PRECOMPUTED IN R0
  882 ;
  883
  884         UNIVERSAL_SYMBOL        SCH$CHSEP
  885 ;SCH$CHSEP::                                ; ENTRY WITH PRIO IN R0
  886         MFPR    #PR$_IPL,R1                 ; GET IPL
  887         CMPL    R1,#IPL$_SYNCH              ; MUST BE AT SYNCH OR GREATER
  888         BLSS    BADIPL                      ; NO, FATAL ERROR
  889         REMQUE  (R4),R1                     ; REMOVE FROM CURRENT QUEUE
  890         BNEQ    10$                         ; CONTINUE IF QUEUE NOTEMPTY
  891         MOVZWL  PCB$W_STATE(R4),R1          ; GET OLD STATE
  892         BBC     R1,EXESTATE,10$             ; NO SUMMARY BITS
  893         MOVZBL  PCB$B_PRI(R4),R1            ; GET CURRENT PRI
  894         BLBC    PCB$W_STATE(R4),5$          ; SKIP IF RESIDENT
  895         ADDL    #32,R1                      ; MAKE NONRES PRIO
  896 5$:     BBCC    R1,G^SCH$GL_COMQS,10$       ; CLEAR PRESENCE BIT FOR STATE
  897 10$:    MOVB    R0,PCB$B_PRI(R4)            ; SAVE NEW PRIO
  898         MOVL    #SCH$C_COM,R1               ; ASSUME COM STATE
  899         BLBS    PCB$L_STS(R4),20$           ; CHECK FOR RESIDENCE
  900         ADDL2   #32,R0                      ; COMO HEADERS FOLLOW COM
  901         BBSS    R0,G^SCH$GL_COMQS,15$       ; SET SUMMARY BIT FOR NEW QUEUE
  902 15$:    BSBW    SCH$SWPWAKE                 ; WAKE SWAPPER
  903         MOVW    #SCH$C_COMO,PCB$W_STATE(R4)     ; SET NEW STATE
  904         MOVAQ   G^SCH$AQ_COMT[R0],R2        ; COMPUTE HDR ADDR
  905         INSQUE  (R4),@(R2)+                 ; INSERT IN NEW QUEUE
  906         RSB                                 ; RETURN
  907
  908 20$:    PUSHR   #^M<R2,R3,R5,R6>
  909 ;
  910 ;       We won't worry if the current affinity mask is out of date in this
  911 ;       path.  If it is missing a CPU, we will quickly spot it in the next
  912 ;       test.  If it has an extra CPU, the scheduler code will quickly fix
  913 ;       things up.
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 22
X-32U5 SCH$CHSE - CHANGE STATE TO EXECUTABLE 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE

  914 ;
  915          MOVL    PCB$L_CURRENT_AFFINITY(R4),R5 ; Get fast copy for multiple tests
  916          BITL    R5,G^SCH$GL_IDLE_CPUS   ; Any eligible, idle CPUs ?
  917          BNEQ    50$                     ; BR if at least one
  918          BICL3   R5,G^SCH$GL_IDLE_CPUS,R2; Any unuseable CPUs
  919          BNEQ    100$                    ; Yes  - go make sure we are up to date
  920          BITL    W^SCH$AL_PREEMPT_MASK[R0],- ; Can we preempt anyone
  921                  G^SCH$GL_ACTIVE_PRIORITY
  922          BEQL    30$                     ; No
  923          ASSUME  SMP$V_ENABLED EQ 0
  924          BLBS    G^SMP$GL_FLAGS,200$     ; No extra checks if uniprocessor
  925 25$:     SOFTINT #IPL$_RESCHED           ; TRIGGER RESCHEDULE INTERRUPT
  926 30$:     BBSS    R0,G^SCH$GL_COMQS,35$   ; SET SUMMARY BIT FOR NEW QUEUE
  927 35$:     MOVW    R1,PCB$W_STATE(R4)      ; SET NEW STATE
  928          MOVAQ   G^SCH$AQ_COMT[R0],R2    ; COMPUTE HDR ADDR
  929          INSQUE  (R4),@(R2)+             ; INSERT IN NEW QUEUE
  930          POPR    #^M<R2,R3,R5,R6>
  931          RSB                             ; RETURN
  932
  933 50$:     BBC     #CPB$V_IMPLICIT_AFFINITY,- ; Is implicit affinity involved
  934                  PCB$L_CAPABILITY(R4),60$
  935          BBSC    PCB$L_AFFINITY(R4),-    ; Yes - see if CPU is idle
  936                  G^SCH$GL_IDLE_CPUS,30$  ; Yes - signal the CPU
  937 60$:     BICL    R5,G^SCH$GL_IDLE_CPUS   ; Signal all eleigible CPUS
  938          BRB     30$
  939
  940 100$:    CMPL    W^SCH$GL_CAPABILITY_SEQUENCE,-  ; Are we up to date
  941                  PCB$L_CAPABILITY_SEQ(R4)
  942          BEQL    110$                            ; Yes
  943          BSBW    SCH$CALCULATE_AFFINITY  ; Redo our mask
  944          MOVL    PCB$L_CURRENT_AFFINITY(R4),R5 ; Get new copy
  945          BITL    R5,G^SCH$GL_IDLE_CPUS   ; Any eligible, idle CPUs ?
  946          BNEQ    50$                     ; BR if at least one
  947 110$:    BITL    W^SCH$AL_PREEMPT_MASK[R0],- ; Can we preempt anyone
  948                  G^SCH$GL_ACTIVE_PRIORITY
  949          BEQL    30$                     ; No
  950          BRB     210$
  951
  952 200$:    CMPL    W^SCH$GL_CAPABILITY_SEQUENCE,-  ; Are we up to date
  953                  PCB$L_CAPABILITY_SEQ(R4)
  954          BEQL    210$                            ; Yes
  955          BSBW    SCH$CALCULATE_AFFINITY  ; Redo our mask
  956          MOVL    PCB$L_CURRENT_AFFINITY(R4),R5 ; Get new copy
  957 210$:    FFS     #0,#32,-                ; Find lowest running priority
  958                  G^SCH$GL_ACTIVE_PRIORITY,R2 ; (external form)
  959          MNEGL   R2,R3
  960          MOVL    W^SCH$AL_CPU_PRIORITY+<4*31>[R3],-
  961                  R3                      ; Get mask of CPUs at this priority
  962          MCOML   R5,R5                   ; Get complement of affinity mask
  963          BICL    R5,R3                   ; Candidate CPUs
  964          BEQL    360$                    ; None
  965 ;
  966 ; At this point we are assured we will find a CPU.
  967 ; If the search for a CPU for which the process can run always
  968 ; starts with the low bit of the candidate CPU longword, the algorithm
  969 ; would be biased towards low CPU numbers. To try to avoid such weighting,
  970 ; the bit corresponding to the CPU on which the transition to COM has
```

```
  971 ; OCCURRED is rotated into the low position, and the search started from
  972 ; there. Beginning the search with the CPU processing the transition has
  973 ; the additional advantage of being biased toward local (as opposed to remote)
  974 ; preemption.
  975 ;
  976 ; In case of implicit affinity, the search will instead be biased towards the
  977 ; CPU for which the process has affinity.
  978 ;
  979 300$:   BBC     #CPB$V_IMPLICIT_AFFINITY,- ; Is implicit affinity involved
  980             PCB$L_CAPABILITY(R4),305$
  981         MNEGL   PCB$L_AFFINITY(R4),R5      ; Get  negative of CPU ID
  982         BRB     308$
  983 305$:   FIND_CPU_DATA   R2
  984         MNEGL   CPU$L_PHY_CPUID(R2),R5  ; ID of this CPU
  985 308$:   ROTL    R5,R3,R3                ; Get our CPU in low bit
  986         BLBC    R3,310$
  987         BRW     25$                     ; we have met him and he is us
  988 310$:   FFS     #0,#32,R3,R3            ; Get  CPU ID
  989         SUBL    R5,R3                   ; Adjust for the ROTL
  990         BICL    #^C<31>,R3              ; MOD 32
  991         IPINT_CPU RESCHED,R3            ; request RESCHED of that CPU
  992 320$:   BRW     30$
  993
  994 360$:   MOVL    G^SCH$GL_ACTIVE_PRIORITY,R6 ; Get currently executing priorities
  995         MCOML   W^SCH$AL_PREEMPT_MASK[R0],R3  ; Complement of the preemption mask
  996         BICL    R3,R6                   ; Priorities we can try to preempt
  997 370$:   INCL    R2                      ; Skip ones  we have looked at
  998         SUBL3   R2,#32,R3               ; Bits to check
  999         FFS     R2,R3,R6,R2             ; Get next candidate CPU ID
 1000         BEQL    320$                    ; Can't run it
 1001         MNEGL   R2,R3
 1002         MOVL    W^SCH$AL_CPU_PRIORITY+<4*31>[R3],-
 1003                 R3                      ; Get mask of CPUs at this priority
 1004         BICL    R5,R3                   ; Candidate CPUs
 1005         BEQL    370$                    ; Try again
 1006         BRW     300$                    ; Got one
 1007

 1008         .SBTTL  SWPO - SWAP OUT SIMPLE NON-EXECUTABLE
 1009 ;
 1010 ;       SWPO - SWAP OUT ACTION ROUTINE FOR SIMPLE NON-EXECUTABLE STATES
 1011 ;
 1012 SWPO:                                   ; NON-EXECUTABLE OUTSWAP
 1013         BSBW    SCH$UNWAIT              ; REMOVE FROM WAIT QUEUE
 1014         INCW    PCB$W_STATE(R4)         ; UPDATE STATE NUMBER
 1015         REMQUE  (R4),R1                 ; REMOVE FROM WAIT QUEUE
 1016         INSQUE  (R4),@WQH$L_WQBL+WQH$C_LENGTH(R0)       ; INSERT AT TAIL OF QUEUE
 1017         INCW    WQH$W_WQCNT+WQH$C_LENGTH(R0)    ; NOTE COUNT IN WAIT QUEUE
 1018         RSB                             ; EXIT
 1019
 1020 ;
 1021 ;       SWPOE - SWAP OUT EXECUTABLE ACTION ROUTINE
 1022 ;
 1023 SWPOE:  MOVZBL  PCB$B_PRI(R4),R0        ; GET PRIORITY
 1024         BRW     SCH$CHSEP               ; AND CHANGE TO COMO
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 24
X-32U5 SCH$WAIT - PLACE PROCESS IN SELECTED WAI 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]R

 1026          .SBTTL  SCH$WAIT - PLACE PROCESS IN SELECTED WAIT QUEUE
 1027
 1028 ;++
 1029 ; FUNCTIONAL DESCRIPTION:
 1030 ;
 1031 ;       SCH$WAIT PLACES THE CURRENT PROCESS IN A WAIT QUEUE
 1032 ;       SELECTED BY A WAIT QUEUE HEADER ADDRESS SUPPLIED IN A REGISTER
 1033 ;       A NEW PROCESS IS THEN SELECTED FOR EXECUTION.
 1034 ;
 1035 ; CALLING SEQUENCE:
 1036 ;
 1037 ;       JMP/BR  SCH$WAIT
 1038 ;
 1039 ; INPUT PARAMETERS:
 1040 ;
 1041 ;       EXE$C_CMSTKSZ+00(SP) - PC AT WHICH TO RESTART PROCESS AFTER EF WAIT
 1042 ;       EXE$C_CMSTKSZ+04(SP) - PSL WITH WHICH TO RESTART PROCESS
 1043 ;
 1044 ; IMPLICIT INPUTS:
 1045 ;
 1046 ;       SCHED DATABASE MUST BE LOCKED, IPL = SYNCH
 1047 ;
 1048 ; SIDE EFFECTS:
 1049 ;
 1050 ;       THE PROCESS SPECIFIED BY THE PCB ADDRESS IN R4 IS PLACED
 1051 ;       IN THE WAIT QUEUE LOCATED BY R2, ITS CONTEXT SAVED,
 1052 ;       AND A NEW PROCESS SCHEDULED.
 1053 ;
 1054 ;--
 1055
 1056          UNIVERSAL_SYMBOL        SCH$WAIT
 1057 ;SCH$WAIT::                                ;PLACE PROCESS IN WAIT STATE
 1058          MOVL    12(SP),FP                 ;RESTORE FP
 1059          ADDL2   S^#EXE$C_CMSTKSZ,SP       ;CLEAN UP KERNEL STACK
 1060          SUBL2   #4,(SP)                   ;BACK UP SAVED PC
 1061          UNIVERSAL_SYMBOL        SCH$WAITK
 1062 ;SCH$WAITK::                               ;WAIT WITH STACK ALREADY CLEAN
 1063          INCW    WQH$W_WQCNT(R2)           ;INCREMENT COUNT FOR QUEUE
 1064          INSQUE  (R4),WQH$L_WQFL(R2)       ;INSERT IN QUEUE
 1065          MOVW    WQH$W_WQSTATE(R2),PCB$W_STATE(R4) ;SET STATE FOR PROCESS
 1066 ;
 1067 ;                                           THE STATE NUMBER IS CONTAINED
 1068 ;                                           IN THE QUEUE HEADER
 1069 ;
 1070          UNIVERSAL_SYMBOL        SCH$WAITL
 1071 ;SCH$WAITL::                                ;WAIT WITH STATE SET, STACK CLEANED
 1072          SVPCTX                            ;SAVE PROCESS CONTEXT
 1073
 1074          UNIVERSAL_SYMBOL        SCH$WAITM
 1075 ;SCH$WAITM::                                ;ENTRY FOR MEMORY MANAGEMENT WAIT CODE
 1076          MOVL    PCB$L_PHD(R4),R5          ;GET PROCESS HEADER ADDRESS
 1077          MOVZWL  G^SCH$GW_IOTA,R0          ;Get a longword copy of IOTA
 1078          ADDW    R0,PHD$W_QUANT(R5)        ;CHARGE QUOTA FOR VOLUNTARY WAIT
 1079          SUBL2   R0,PHD$L_TIMREF(R5)       ;Adjust AWSA time reference
 1080          MOVL    G^EXE$GL_ABSTIM_TICS,PCB$L_WAITIME(R4) ;RECORD TIME AT WAIT START
 1081          CMPB    #4,PHD$B_ASTLVL(R5)       ;NULL ASTLVL?
 1082          BNEQ    20$                       ;NO, DO LONG CHECK
```

```
1083 10$:    BRW     SCH$SCHED               ;GO SCHEDULE NEXT PROCESS
1084
1085 20$:    MOVZBL  PHD$B_ASTLVL(R5),R0     ;FETCH AND ZERO EXTEND PENDING ASTLVL
1086         CMPZV   #PSL$V_CURMOD,#PSL$S_CURMOD,PHD$L_PSL(R5),R0 ;COMPARE WITH WAIT
1087                                         ;ACCESS MODE
1088         BLSS    10$                     ;BRANCH IF AST NOT DELIVERABLE
1089
1090 ;
1091 ;  Test for assumptions that are being made about the layout of the
1092 ;  PSL that enables the next instruction to work correctly.
1093 ;
1094 ;       o  IPL field begins on a byte boundary
1095 ;       o  IPL field fits into a single byte
1096
1097
1098         ASSUME  <<<PSL$V_IPL/8>*8> - PSL$V_IPL> EQ 0 ; IPL must be byte aligned
1099         ASSUME  PSL$S_IPL LE 8          ; IPL field must fit into byte
1100
1101         BITB    #<PSL$M_IPL@-PSL$V_IPL>,-
1102                 <PSL$V_IPL/8>+PHD$L_PSL(R5) ;MUST BE AT IPL 0 FOR DELIVERY
1103         BNEQ    10$                     ;BRANCH IF AST NOT DELIVERABLE
1104         CLRL    R2                      ;SET NULL PRIORITY INCREMENT
1105         RPTEVT  AST                     ;REPORT AST EVENT
1106         BRB     10$                     ;GO SCHEDULE NEXT PROCESS
1107
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 26
X-32U5 SCH$PIXSCAN - Process priority aging 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MP

 1109          .SBTTL  SCH$PIXSCAN - Process priority aging
 1110
 1111 ;++
 1112 ; FUNCTIONAL DESCRIPTION:
 1113 ;
 1114 ;       SCH$PIXSCAN is invoked to give selected, COM/COMO
 1115 ;       processes a priority boost.
 1116 ;
 1117 ; CALLING SEQUENCE:
 1118 ;
 1119 ;       JSB     SCH$PIXSCAN
 1120 ;
 1121 ; INPUT PARAMETERS:
 1122 ;
 1123 ;       NONE
 1124 ;
 1125 ; IMPLICIT INPUTS:
 1126 ;
 1127 ;       IPL = SYNCH
 1128 ;
 1129 ; SIDE EFFECTS:
 1130 ;
 1131 ;       R0-R6 destroyed.
 1132 ;       Selected processes have received a priority boost.
 1133 ;
 1134 ;--
 1135
 1136 LOWPRI: .BYTE   31,30
 1137
 1138          UNIVERSAL_SYMBOL        SCH$PIXSCAN
 1139 ;SCH$PIXSCAN::
 1140          BBS     S^#SCH$V_NO_PIXSCAN,-   ; BR if process aging with this
 1141                  W^SCH$$GL_CTLFLAGS,5$   ; mechanism is disabled
 1142          MOVZWL  G^SGN$GW_PIXSCAN,R5     ; Get max no. of processes to boost
 1143          BNEQ    10$                     ; BR if not disabled
 1144 5$:     RSB
 1145
 1146 7$:     BRW     100$                    ; Branch helper
 1147
 1148 10$:    PUSHR   #^M<R7,R8,R9,R10,R11>   ; Save working registers
 1149
 1150          LOCK    LOCKNAME=SCHED,-        ; Acquire scheduler lock
 1151                  LOCKIPL=#IPL$_SYNCH,-
 1152                  CONDITION=NOSETIPL,-
 1153                  PRESERVE=NO             ; OK to destroy R0
 1154
 1155          BISL3   G^SCH$GL_COMQS,-        ; Merge COM/COMO summary longwords
 1156                  G^SCH$GL_COMOQS,R11
 1157          BICL2   #^X8000FFFF,R11         ; Mask NULL and real-time processes
 1158          BEQL    7$                      ; BR if no processes are COM/COMO
 1159
 1160          BSBW    200$                    ; Merge priority(ies) of CUR process(es)
 1161
 1162          FFS     #16,#15,R11,R11         ; Find highest priority, non-realtime,
 1163                                          ; CUR, COM or COMO process
 1164                                          ; (ignoring external priority 0)
 1165                                          ; (If none, R11=31; will terminate scan)
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 27
X-32U5 SCH$PIXSCAN - Process priority aging 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MP

 1166
 1167          EXTZV   S^#SCH$V_PIXSCAN_LOWPRI,-; Determine if (external) priority 0
 1168                  #1,W^SCH$$GL_CTLFLAGS,R6 ; processes receive a priority boost
 1169          MOVZBL  B^LOWPRI[R6],R3          ; R3 = highest (internal) priority to check
 1170
 1171          EXTZV   #0,#1,G^EXE$GL_ABSTIM,R9; Initialize scan state flags
 1172
 1173          MOVZWL  G^SCH$GW_DORMANTWAIT,R10; Convert DORMANTWAIT seconds
 1174          MULL2   #100,R10                 ; to 10 ms units for comparisons
 1175
 1176                                           ; With a 50/50 probability, decide on
 1177                                           ; whether to start with COM or COMO
 1178                                           ; processes.
 1179
 1180 25$:    BLBS    R9,50$                   ; BR if we should do COMO 1st this pass
 1181
 1182 30$:    MOVAQ   G^SCH$AQ_COMH,R7         ; Setup to process COM processes
 1183          MOVL    G^SCH$GL_COMQS,R8
 1184          BRB     70$
 1185
 1186 40$:    XORL2   #2,R9                    ; Toggle scan state flag
 1187          BBS     #1,R9,45$                ; BR if scanning same priority
 1188          DECL    R3                       ; Setup for next priority
 1189          BRB     25$                      ; Go scan for processes
 1190
 1191 45$:    BLBS    R9,30$                   ; BR if we did COMO first, now do COM
 1192
 1193 50$:    MOVAQ   G^SCH$AQ_COMOH,R7        ; Setup to process COMO processes
 1194          MOVL    G^SCH$GL_COMOQS,R8
 1195 ;        BRB     70$
 1196
 1197 70$:    CMPL    R3,R11                   ; Are we finished ?
 1198          BLEQU   100$                     ; BR if yes
 1199          BBC     R3,R8,40$                ; Look for a non-empty queue
 1200
 1201          MOVAQ   (R7)[R3],R6              ; R6 = address of queue header
 1202          MOVL    (R6),R4                  ; R4 = address of 1st PCB in queue
 1203          BRB     90$
 1204
 1205 80$:    CMPL    R4,R6                    ; Reached end of list ?
 1206          BEQL    40$                      ; Yes, try higher priority processes or
 1207                                           ; switch COM/COMO processing context
 1208
 1209 90$:    ADDL3   PCB$L_ONQTIME(R4),-      ; Has this process been waiting for the
 1210                  R10,R0                   ; CPU a long time ?
 1211          CMPL    G^EXE$GL_ABSTIM_TICS,R0
 1212          BLSSU   100$                     ; No, quit boosting processes
 1213
 1214          BISL2   #1,PCB$L_PIXHIST(R4)     ; Show priority boost given
 1215          ASSUME  PCB$L_SQFL EQ 0
 1216          PUSHL   (R4)                     ; Save address of next PCB in queue
 1217          MOVL    R11,R0                   ; Desired priority must be in R0
 1218          BSBW    SCH$CHSEP                ; Give temporary priority boost
 1219          POPL    R4                       ; Get next potential candidate process
 1220          DECL    R5                       ; Maximum number of processes boosted ?
 1221          BLEQ    100$                     ; BR if yes
 1222          BRB     80$                      ; Else examine more processes
```

```
1223
1224 100$:   UNLOCK LOCKNAME=SCHED,-          ; Release scheduler lock
1225                 PRESERVE=NO              ; OK to destroy R0
1226         POPR    #^M<R7,R8,R9,R10,R11>    ; Restore working registers
1227         RSB
1228
1229 ;
1230 ; Local subroutine to merge priorities of CURrent processes on all CPUs
1231 ;
1232
1233 200$:   BLBS    G^SMP$GL_FLAGS,205$      ; BR if multiprocessor configuration
1234         FIND_CPU_DATA   R1
1235         CVTBL   CPU$B_CUR_PRI(R1),R1     ; R1 = Priority of current process
1236         BLSS    202$                     ; Negative priority means CPU is idle
1237         BBSS    R1,R11,202$              ; Merge current process' priority
1238
1239 202$:   BICL2   #^X8000FFFF,R11          ; Mask NULL and real-time processes
1240         RSB                              ; and return with updated bit mask
1241
1242 205$:   LOCK    MUTEX=SMP$GL_CPU_MUTEX,-; Protect active CPU set
1243                 SHARE=YES,-
1244                 PRESERVE=NO
1245         CLRL    R0                       ; R0 = starting bit in active CPU mask
1246         MOVL    G^SMP$GL_ACTIVE_CPUS,R2  ; R2 = copy of in memory active CPU mask
1247         MOVAL   G^SMP$GL_CPU_DATA,R3     ; R3 = addr of CPU database addr table
1248 210$:   SUBL3   R0,#32,R1                ; R1 = # bits remaining to scan
1249         FFS     R0,R1,R2,R0              ; Find first active CPU
1250         BEQL    230$                     ; BR if none left
1251         MOVL    (R3)[R0],R1              ; R1 = address of database for this CPU
1252         CVTBL   CPU$B_CUR_PRI(R1),R1     ; R1 = Priority of current process
1253         BLSS    220$                     ; Negative priority means CPU is idle
1254         BBSS    R1,R11,220$              ; Merge current process' priority
1255 220$:   AOBLSS  #32,R0,210$              ; R0 points to next starting position
1256 230$:   UNLOCK  MUTEX=SMP$GL_CPU_MUTEX,-; Release active CPU mutex
1257                 SHARE=YES,-
1258                 PRESERVE=NO
1259         BRW     202$
1260
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 29
X-32U5 SCH$ONE_SEC - Perform periodic, 1 second 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RS

 1262          .SBTTL  SCH$ONE_SEC - Perform periodic, 1 second processing
 1263
 1264 ;++
 1265 ; FUNCTIONAL DESCRIPTION:
 1266 ;
 1267 ;       SCH$ONE_SEC is invoked every second by EXE$TIMEOUT to perform
 1268 ;       any periodic processing required for the CPU scheduling subsystem.
 1269 ;
 1270 ; CALLING SEQUENCE:
 1271 ;
 1272 ;       JSB     SCH$ONE_SEC
 1273 ;
 1274 ; INPUT PARAMETERS:
 1275 ;
 1276 ;       NONE
 1277 ;
 1278 ; IMPLICIT INPUTS:
 1279 ;
 1280 ;       IPL = SYNCH
 1281 ;
 1282 ; SIDE EFFECTS:
 1283 ;
 1284 ;       R0-R6 destroyed.
 1285 ;
 1286 ;--
 1287
 1288 ;
 1289 ; Strictly speaking, the word references to SCH$GW_IOTA and SCH$GW_QUAN
 1290 ; should be interlocked. However, the probability of their being changed
 1291 ; concurrently (or at all) approaches 0, and the timing window is only
 1292 ; a few instructions. Since the results of a wrong decision are harmless
 1293 ; and will be corrected momentarily, we'll take our chances !!!
 1294 ;
 1295          UNIVERSAL_SYMBOL        SCH$ONE_SEC
 1296 ;SCH$ONE_SEC::
 1297 ;       MOVW    #MINWC,R0               ; Set working MINWC to maximum of
 1298 ;       CMPW    R0,G^SCH$GW_IOTA        ; nominal MINWC and IOTA
 1299 ;       BGEQU   5$
 1300 ;       MOVW    G^SCH$GW_IOTA,R0
 1301 ;5$:    MOVZWL  R0,W^SCH$GL_MINWC
 1302 ;
 1303 ;       BICB3   #^C<15>,G^SCH$GB_MINCLASSPRI,R0 ; Update local copy of MINCLASSPRI
 1304 ;       SUBB3   R0,#31,W^SCH$$GB_MINCLASSPRI ;  (internal priority form)
 1305 ;
 1306 ;       BICB3   #^C<15>,G^SCH$GB_MAXCLASSPRI,R0 ; Update local copy of MAXCLASSPRI
 1307 ;       SUBB3   R0,#31,W^SCH$$GB_MAXCLASSPRI ;  (internal priority form)
 1308 ;
 1309 ;       BICB3   #^C<15>,G^SCH$GB_MINPRPRI,R0 ; Update local copy of MINPRPRI
 1310 ;       SUBB3   R0,#31,W^SCH$$GB_MINPRPRI ;  (internal priority form)
 1311 ;
 1312 ;       MOVL    G^SCH$GL_CTLFLAGS,R1    ; R1 = working copy of system parameter
 1313 ;       CMPL    W^SCH$$GL_CTLFLAGS,-    ; Local copy of control flags same
 1314 ;               R1                      ; as system parameter ?
 1315 ;       BNEQ    15$                     ; BR if any flags are different
 1316 7$:     ADDW3   W^SCH$$GW_QUAN,-        ; Has basic quantum value
 1317                 G^SCH$GW_QUAN,R0        ; changed since last time ?
 1318 ;+++ TEMP +++
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 30
X-32U5 SCH$ONE_SEC - Perform periodic, 1 second 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RS

 1319 ;
 1320          BEQL    10$                     ; BR if no
 1321          MNEGW   G^SCH$GW_QUAN,-         ; Update local copy of basic quantum
 1322                  W^SCH$$GW_QUAN
 1323 ;
 1324 ;--- TEMP ---
 1325 ;        BNEQ    17$                     ; BR if yes
 1326 10$:     BSBW    SCH$PIXSCAN             ; Do process priority aging
 1327          RSB
 1328
 1329 ;15$:    MOVL    R1,W^SCH$$GL_CTLFLAGS   ; Update local copy of control flags
 1330 ;        BITL    #<SCH$M_PRI_QUANT!SCH$M_WAIT_CREDIT>,-
 1331 ;                R1                      ; Take easy way out if wait credit and
 1332 ;        BEQL    7$                      ; queue dependent quantums both disabled
 1333 ;        BRB     20$
 1334 ;
 1335 ;17$:    MNEGW   G^SCH$GW_QUAN,-         ; Update local copy of basic quantum
 1336 ;                W^SCH$$GW_QUAN
 1337 ;        BITL    #<SCH$M_PRI_QUANT!SCH$M_WAIT_CREDIT>,-
 1338 ;                R1                      ; Take easy way out if wait credit and
 1339 ;        BEQL    10$                     ; queue dependent quantums both disabled
 1340 ;
 1341 ;20$:    LOCK    LOCKNAME=SCHED,-        ; Acquire scheduler lock
 1342 ;                LOCKIPL=#IPL$_SYNCH,-
 1343 ;                CONDITION=NOSETIPL,-
 1344 ;                PRESERVE=NO             ; OK to destroy R0
 1345 ;        MOVL    R1,R0                   ; Update working copy of control flags
 1346 ;        MOVL    R0,W^SCH$$GL_CTLFLAGS
 1347 ;        MNEGW   G^SCH$GW_QUAN,R6        ; Update working copy of basic quantum
 1348 ;;       ASSUME  SCH$V_WAIT_CREDIT EQ SCH$V_PRI_QUANT+1
 1349 ;        EXTZV   S^#SCH$V_PRI_QUANT,#2,R0,R1; Form index for table lookup
 1350 ;        MOVZWL  W^SCH$GW_MAX_QUANT[R1],R5; R5 = maximum allowed quantum value
 1351 ;        CMPW    R6,R5                   ; Quantum value in range ?
 1352 ;        BLEQ    25$                     ; BR if yes
 1353 ;        MOVW    R5,R6                   ; No, use maximum value
 1354 ;        MNEGW   R6,G^SCH$GW_QUAN        ; Reset system copy
 1355 ;25$:    MOVW    R6,W^SCH$$GW_QUAN       ; Update local copy of basic quantum
 1356 ;
 1357 ; Populate quantum and wait credit tables as a function of the base quantum
 1358 ; value and whether the quantums should be queue dependent.
 1359 ;
 1360 ;        MOVL    #MAX_BOOST-1,R1         ; Initialize loop control
 1361 ;        MOVAW   W^SCH$GW_MAXWC+2[R1],R2 ; Set table end addresses for loop
 1362 ;        MOVAW   W^SCH$GW_QUANTUM+2[R1],R3
 1363 ;        MOVZWL  R6,R6                   ; Get a LW copy of the basic quantum
 1364 ;30$:    ADDW3   (R2),(R3),-(R2)         ; Set accumulated wait credit for queue
 1365 ;        MOVL    R6,R4                   ; Assume queue-independent quantum
 1366 ;        BBC     S^#SCH$V_PRI_QUANT,R0,35$; BR if correct assumption
 1367 ;        MULL2   W^SCH$GL_QUANT_MULT[R1],R4; Scale by queue-dependent multiplier
 1368 ;        CMPL    R4,R5                   ; Quantum within allowable range ?
 1369 ;        BLEQ    35$                     ; BR if within range
 1370 ;        MOVL    R5,R4                   ; Limit quantum to maximum value
 1371 ;35$:    MOVW    R4,-(R3)                ; Store resulting quantum in table
 1372 ;        SOBGEQ  R1,30$                  ; Loop to setup table for all queues
 1373 ;        ADDW3   (R2),(R3),R1            ; Compute accumulated wait credit for
 1374 ;        MOVZWL  R1,W^SCH$GL_MAXWC       ; entire table and store LW copy
 1375 ;        UNLOCK LOCKNAME=SCHED,-         ; Release scheduler lock
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 31
X-32U5 SCH$ONE_SEC - Perform periodic, 1 second 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RS

 1376 ;                 PRESERVE=NO             ; OK to destroy R0
 1377 ;         BRW     10$
 1378 ;
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 32
X-32U5 SCH$QEND - QUANTUM END ROUTINE 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (1

 1380          .SBTTL  SCH$QEND - QUANTUM END ROUTINE
 1381
 1382 ;++
 1383 ;
 1384 ; FUNCTIONAL DESCRIPTION:
 1385 ;       SCH$QEND IS CALLED BY THE TIMER WHEN THE QUANTUM FOR THE CURRENT
 1386 ;       PROCESS HAS BEEN EXHAUSTED.  A NEW QUANTUM IS INITIALIZED
 1387 ;       THE PROCESS PLACED AT ITS BASE PRIORITY AND THE RESCHEDULING
 1388 ;       INTERRUPT TRIGGERED.  A CHECK IS MADE FOR CPU TIME LIMIT EXPIRATION
 1389 ;       AND APPROPRIATE EXIT ASTS GENERATED WHEN THE LIMIT IS REACHED.
 1390 ;       THE AUTOMATIC WORKING SET SIZE LOGIC IS INVOKED IF ENABLED TO
 1391 ;       TRADEOFF WORKING SET SIZE AGAINST PAGEFAULT RATE.
 1392 ;
 1393 ; CALLING SEQUENCE:
 1394 ;       BSB/JSB G^SCH$QEND
 1395 ;
 1396 ; INPUT PARAMETERS:
 1397 ;       R4 - PCB ADDRESS OF CURRENT PROCESS
 1398 ;       R5 - PROCESS HEADER ADDRESS
 1399 ;
 1400 ; IMPLICIT INPUTS:
 1401 ;       PCB OF CURRENT PROCESS
 1402 ;       PROCESS HEADER OF CURRENT PROCESS
 1403 ;
 1404 ; IMPLICIT OUTPUTS:
 1405 ;       PHD$W_QUANT - INITIALIZED TO A NEW QUANTUM
 1406 ;       PCB$V_INQUAN - INITIAL QUANTUM FLAG CLEARED
 1407 ;
 1408 ; ENVIRONMENT:
 1409 ;       IPL = IPL$_SYNCH, SCHED DATABASE LOCKED
 1410 ;--
 1411
 1412          UNIVERSAL_SYMBOL        SCH$QEND
 1413 ;SCH$QEND::                                ; QUANTUM END ROUTINE
 1414          BBCC    #PCB$V_INQUAN,PCB$L_STS(R4),10$ ; CLEAR INITIAL QUAN FLAG
 1415 10$:     MNEGW   W^SCH$$GW_QUAN,-          ; Set new quantum (Assume that queue
 1416                  PHD$W_QUANT(R5)           ; dependent quantums are disabled.)
 1417          MOVL    G^EXE$GL_ABSTIM_TICS,-    ; Record time placed on COM queue
 1418                  PCB$L_ONQTIME(R4)
 1419          CMPB    PCB$B_PRI(R4),#16         ; CHECK FOR REAL-TIME
 1420          BGEQ    20$
 1421          RSB
 1422 20$:     ASHL    #1,PCB$L_PIXHIST(R4),-    ; Update PIXSCAN history summary LW
 1423                  PCB$L_PIXHIST(R4)
 1424 ;
 1425 ;
 1426 ;       CHECK FOR CPU TIME LIMIT EXPIRATION
 1427 ;
 1428          TSTL    PHD$L_CPULIM(R5)          ; IS THERE ANY LIMIT?
 1429          BEQL    40$                       ; BR if no
 1430          BRW     60$                       ; Else, go check it out
 1431 40$:     BBS     #PCB$V_DISAWS,PCB$L_STS(R4),45$ ; BRANCH IF ADJUSTMENT DISABLED
 1432          BBS     #PHD$V_NO_WS_CHNG,-       ; WORKING SET CHANGES NOT ALLOWED
 1433                  PHD$W_FLAGS(R5),45$
 1434          MOVL    G^SCH$GL_WSINC,R3         ; ASSUME INCREMENT
 1435          BEQL    45$                       ; BR IF NO AUTO WS ADJUSTMENT
 1436          BSBW    WSADJUST                  ; ELSE GO DO IT
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 33
X-32U5 SCH$QEND - QUANTUM END ROUTINE 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (1

 1437 45$:    FIND_CPU_DATA   R3
 1438         MOVZBL  PCB$B_PRI(R4),R1        ; CURRENT PRIORITY - INTERNAL FORM
 1439         TSTL    G^SCH$GL_COMQS          ; IS THERE ANY INSWAP PENDING?
 1440         BEQL    47$                     ; NO
 1441         MOVB    PCB$B_PRIB(R4),PCB$B_PRI(R4)    ; YES, FORCE TO BASE PRIORITY
 1442         BICL    CPU$L_CPUID_MASK(R3),-  ;GET MASK FOR CURRENT CPU ID
 1443                 W^SCH$AL_CPU_PRIORITY[R1] ;CLEAR CPU BIT
 1444         BNEQ    46$                     ;ANYONE ELSE AT THIS PRIORITY?
 1445         SUBL3   R1,#31,R1               ;GET PRIORITY IN EXTERNAL FORMAT
 1446         BBCC    R1,-                    ;NO ONE ELSE
 1447                 G^SCH$GL_ACTIVE_PRIORITY,46$
 1448 46$:    MOVZBL  PCB$B_PRI(R4),R1        ; NEW PRIORITY - INTERNAL FORM
 1449         MOVB    R1,CPU$B_CUR_PRI(R3)    ; NEW PRIORITY FOR THIS CPU
 1450         BISL    CPU$L_CPUID_MASK(R3),-  ;GET MASK FOR CURRENT CPU ID
 1451                 W^SCH$AL_CPU_PRIORITY[R1] ;SET CPU BIT
 1452         SUBL3   R1,#31,R2               ;GET PRIORITY IN EXTERNAL FORMAT
 1453         BBSS    R2,-
 1454                 G^SCH$GL_ACTIVE_PRIORITY,49$
 1455 49$:    PUSHAB  B^48$
 1456         BRW     SCH$SWPWAKE             ; AND WAKE SWAPPER
 1457 47$:;   BICL3   #^C<SCH$M_QEND_PRIADJ!SCH$M_NO_QEND_PREEMPT!SCH$M_PRI_QUANT>,-

 1458 ;               W^SCH$$GL_CTLFLAGS,R0   ; Fetch scheduling control flags to test
 1459 ;       BNEQ    55$
 1460         ASSUME  PCB$V_QUANTUM_RESCHED LE 7
 1461 48$:    FFS     #0,#32,-                ; Find highest COM priority
 1462                 G^SCH$GL_COMQS,R2
 1463         CMPL    R1,R2                   ; Check against current priority
 1464         BGEQ    51$                     ; Resched if equal or greater
 1465         CMPB    R1,PCB$B_PRIB(R4)       ; Are we at base
 1466         BGEQ    50$                     ; Don't adjust - should never be GTR
 1467         ADDL3   #1,R1,R0                ; Move priority towards base
 1468         MOVL    R3,R1                   ; Needed for next routine
 1469         BSBW    SCH$CHANGE_CUR_PRIORITY
 1470 50$:    RSB
 1471
 1472 51$:    BISB2   #PCB$M_QUANTUM_RESCHED,-
 1473                 PCB$L_STS2(R4)          ; Show quantum-oriented reschedule
 1474         SOFTINT #IPL$_RESCHED           ; TRIGGER RESCHEDULING INT
 1475         RSB                             ; AND RETURN
 1476
 1477 ;55$:   BBC     S^#SCH$V_QEND_PRIADJ,-  ; BR if quantum end should not
 1478 ;               R0,56$                  ; cause a priority adjustment
 1479 ;
 1480 ;       CMPB    PCB$B_PRIB(R4),-        ; Process already at or below
 1481 ;               PCB$B_PRI(R4)           ; base priority ?
 1482 ;       BLEQU   56$                     ; BR if yes
 1483 ;       INCB    PCB$B_PRI(R4)           ; Decrement process priority
 1484 ;       MOVB    PCB$B_PRI(R4),-         ; Announce new priority
 1485 ;               CPU$B_CUR_PRI(R3)
 1486 ;       BICL    CPU$L_CPUID_MASK(R3),-  ;GET MASK FOR CURRENT CPU ID
 1487 ;               W^SCH$AL_CPU_PRIORITY[R1] ;CLEAR CPU BIT
 1488 ;       BNEQ    51$                     ;ANYONE ELSE AT THIS PRIORITY?
 1489 ;       SUBL3   R1,#31,R1               ;GET PRIORITY IN EXTERNAL FORMAT
 1490 ;       BBCC    R1,-                    ;NO ONE ELSE
 1491 ;               G^SCH$GL_ACTIVE_PRIORITY,51$
 1492 ;51$:           MOVZBL  PCB$B_PRI(R3),R1        ; NEW PRIORITY - INTERNAL FORM
 1493 ;       BISL    CPU$L_CPUID_MASK(R3),-  ;GET MASK FOR CURRENT CPU ID
```

```
1494 ;                  W^SCH$AL_CPU_PRIORITY[R1] ;SET CPU BIT
1495 ;          SUBL3   R1,#31,R1               ;GET PRIORITY IN EXTERNAL FORMAT
1496 ;          BBSS    R1,-
1497 ;                  G^SCH$GL_ACTIVE_PRIORITY,56$
1498 ;
1499 ;56$:     BBC     S^#SCH$V_PRI_QUANT,-    ; BR if not using queue dependent
1500 ;                  R0,57$                  ; quantum values
1501 ;          SUBB3   PCB$B_PRI(R4),-         ; Compute priority relative to base
1502 ;                  PCB$B_PRIB(R4),R3
1503 ;          BLSS    57$                     ; BR is current priority < base
1504 ;          CMPB    #MAX_BOOST,R3           ; Is current priority > max boost ?
1505 ;          BLSS    57$                     ; BR if yes
1506 ;          MOVZBL  R3,R3                   ; Get a LW copy for indexing
1507 ;          MNEGW   W^SCH$GW_QUANTUM[R3],-  ; Set correct quantum value for queue
1508 ;                  PHD$W_QUANT(R5)
1509 ;
1510 ;57$:     BBC     S^#SCH$V_NO_QEND_PREEMPT,- ; BR if non-preeptive reschedule
1511 ;                  R0,58$                  ; policy not enabled
1512 ;          CMPB    PCB$B_PRIB(R4),-        ; Process priority high enough ?
1513 ;                  W^SCH$$GB_MINPRPRI
1514 ;          BGEQU   58$                     ; BR if not
1515 ;          BBC     S^#SCH$V_QEND_PRIADJ,-  ; BR if priority not manipulated here
1516 ;                  R0,59$
1517 ;          FFS     #0,#32,G^SCH$GL_COMQS,R0 ; Find highest priority COM process
1518 ;          CMPB    PCB$B_PRI(R4),R0        ; If current process is still (strictly)
1519 ;          BLSS    59$                     ; higher priority, avoid reschedule
1520 ;58$:
1521 ;          ASSUME  PCB$V_QUANTUM_RESCHED LE 7
1522 ;          BISB2   #PCB$M_QUANTUM_RESCHED,-
1523 ;                  PCB$L_STS2(R4)          ; Show quantum-oriented reschedule
1524 ;          SOFTINT #IPL$_RESCHED           ; Trigger reschedule interrupt
1525 ;59$:     RSB
1526
1527 ;
1528 ;          A non-zero limit exists, check for processor time expiration
1529 ;
1530 ;          If CPU time limit is exceeded then an additional amount of time will
1531 ;          be allowed for each access mode.  An AST will be issued to cause an
1532 ;          exit for each of the access modes.  The additional time allowance will
1533 ;          be provided for each access mode.
1534 ;
1535
1536 60$:      SUBL3   PHD$L_CPULIM(R5),PHD$L_CPUTIM(R5),R0 ; HAS LIMIT BEEN REACHED
1537          BGEQU   65$                     ; BR if yes
1538          BRW     40$                     ; Else, continue normally
1539 ;
1540 ;          CPU LIMIT HAS EXPIRED, AN AST WILL BE SENT TO NOTIFY THE PROCESS
1541 ;
1542 65$:      ADDL2   G^SGN$GL_EXTRACPU,R0    ; COMPUTE TOTAL AMOUNT OF EXTRA TIME
1543          ADDL2   R0,PHD$L_CPULIM(R5)     ; GIVE EXTRA TIME FOR CLEANUP
1544          ADDL2   R0,PHD$L_EXTRACPU(R5)   ;  AND RECORD AMOUNT OF EXTRA TIME
1545          MOVAB   PHD$B_CPUMODE(R5),R0    ; GET ADDRESS OF AST ACCESS MODE
1546          PUSHAB  W^40$                   ; SET RETURN ADDRESS
1547          MOVZWL  #SS$_EXCPUTIM,R3        ; PASS EXIT STATUS TO SENDAST
1548
1549          UNIVERSAL_SYMBOL        SCH$FORCEDEXIT
1550 ;SCH$FORCEDEXIT::
```

```
1551          BSBW    SENDAST                 ; SEND AST TO PROCESS
1552 ;
1553 ;        CPU TIME EXPIRATION AST HANDLER
1554 ;
1555 CPUABRT:.WORD    0                       ; NULL ENTRY MASK
1556          CHMK    S^#ASTEXIT              ; EXIT FROM AST ROUTINE (CLEAR AST)
1557 10$:     $EXIT_S 4(AP)                   ; EXIT TO INVOKE EXIT HANDLERS
1558          BRB     10$                     ; JUST IN CASE
1559
1560 ;
1561 ;        Adjust working set size automatically to achieve desired tradeoff
1562 ;        between page fault rate and working set size.  There are two page
1563 ;        fault rate thresholds: G^SCH$GL_PFRATL, the lower threshold and
1564 ;        SCH$GL_PFRATH, the higher threshold.  Each time G^SCH$QEND is invoked,
1565 ;        the page fault rate is computed and compared with these thresholds.
1566 ;        If it is above the high threshold the working set size is increased
1567 ;        by SCH$GW_WSINC and if the rate is below the lower threshold, the
1568 ;        working set size is decreased by SCH$GW_WSDEC.  The actual adjustment
1569 ;        is performed by a normal kernel mode AST.
1570 ;
1571 ;        Automatic adjustment of working set size is constrained by the values:
1572 ;        SCH$GL_AWSMIN and WSEXTENT per process that establish upper and lower
1573 ;        values  for automatic working set size adjustment.  Working set size
1574 ;        adjustment is further constrained by the process quota.
1575 ;
1576 ;
1577 ;        R3 - Working set increment
1578 ;
1579 WSADJUST:                               ; AUTO-ADJUST WORKING SET SIZE
1580          SUBL3   PHD$L_TIMREF(R5),-     ; COMPUTE DELTA-T
1581                  PHD$L_CPUTIM(R5),R0    ;
1582          BNEQ    10$                    ; BR IF NON-ZERO
1583          INCL    R0                     ; ELSE FORCE TO ONE FOR DIVIDE
1584 10$:
1585          CMPL    R0,G^SCH$GL_AWSTIME    ; IS THIS A MEANINGFUL INTERVAL?
1586          BLSS    20$                    ; BR if no
1587          SUBL3   PHD$L_PFLREF(R5),-     ; COMPUTE DELTA-PGFLT
1588                  PHD$L_PAGEFLTS(R5),R1  ;
1589          MOVL    PHD$L_PAGEFLTS(R5),PHD$L_PFLREF(R5) ; SAVE NEW PAGE FAULT REF
1590          MOVL    PHD$L_CPUTIM(R5),PHD$L_TIMREF(R5)   ; AND SAVE CPUTIME REF
1591          MULL    #1000,R1               ; MULTIPLY BY SCALE FACTOR
1592          DIVL    R0,R1                  ; AND COMPUTE PAGEFLTS/10SEC
1593          MOVL    R1,PHD$L_PFLTRATE(R5)  ; SAVE CURRENT RATE
1594          CMPL    R1,G^SCH$GL_PFRATH     ; ARE WE ABOVE HIGH THRESHOLD?
1595          BGEQ    ADJUSTUP               ; YES,
1596          CMPL    PCB$L_PPGCNT(R4),G^SCH$GL_AWSMIN ; ARE WE AT LOWER WS LIMIT?
1597          BLEQU   NOADJUST               ; YES, NOTHING TO DO
1598 20$:     MNEGL   G^SCH$GL_WSDEC,R3      ; Get decrement value
1599          BEQL    NOADJUST               ; BR if WS decrementing disabled
1600          MOVL    G^SCH$GL_PFRATL,R2     ; R2 = lower PF rate threshold
1601          BEQL    CHECK_WSSIZE           ; BR if possible WS reduction based on free
1602          CMPL    R0,G^SCH$GL_AWSTIME    ; IS THIS A MEANINGFUL INTERVAL?
1603          BLSS    NOADJUST               ; BR if no
1604          CMPL    R1,R2                  ; ARE WE BELOW LOW THRESHOLD?
1605          BLSS    ADJUST                 ; BR if yes
1606 NOADJUST:                              ;
1607          RSB                            ;
```

```
1608 ;
1609 ; PFRATL is known to be zero, i.e., swapper trimming
1610 ; is the preferred memory reclaimation mechanism.
1611 ;
1612 ; WSDEC is known to be non-zero, indicating that it is valid to attempt
1613 ; to shrink process working sets here if memory is currently at a premium,
1614 ; the process has extended its working set beyond WS quota, and the process
1615 ; PIXSCAN history indicates that it is a low-priority, background process.
1616 ;
1617 CHECK_WSSIZE:
1618         TSTL    PCB$L_PIXHIST(R4)       ; Possible candidate based upon
1619                                         ; PIXSCAN history summary LW ?
1620         BEQL    NOADJUST                ; BR is not
1621         CMPL    G^SCH$GL_FREECNT,-      ; Available memory still allow pages to
1622                 G^SCH$GL_GROWLIM        ; be added to extended working sets ?
1623         BGEQU   NOADJUST                ; BR if yes
1624         SUBL3   PHD$L_WSLIST(R5),-
1625                 PHD$L_WSQUOTA(R5),R1    ; R1 = current WSQUOTA
1626         INCL    R1
1627         SUBL2   PHD$L_WSSIZE(R5),R1     ; R1 = -(# pages above WSQUOTA)
1628         BGEQU   NOADJUST                ; BR if WSSIZE still below quota
1629         CMPL    R1,R3                   ; Use smaller of WSDEC and
1630         BLEQU   10$                     ; WS size overage
1631         MOVL    R1,R3                   ; as amount to reduce WS size
1632 10$:    BRB     ADJUST                  ; Reclaim some pages beyond WSQUOTA
1633
1634 ADJUSTUP:                               ;
1635         SUBL3   PHD$L_WSLIST(R5),PHD$L_WSQUOTA(R5),R1
1636                                         ; ASSUME HIGH LIMIT WILL BE QUOTA
1637         MOVL    PHD$L_WSSIZE(R5),R0     ; GET CURRENT WORKING SET SIZE
1638         CMPL    G^SCH$GL_BORROWLIM,G^SCH$GL_FREECNT ; ARE THERE LOTS OF FREE PAGES?
1639         BGTRU   10$                     ; BRANCH IF MEMORY IS AT A PREMIUM
1640         SUBL3   PHD$L_WSLIST(R5),PHD$L_WSEXTENT(R5),R1
1641                                         ; ALLOW LARGER GROWTH SIZE
1642 10$:    CMPL    R0,R1                   ; ARE WE AT MAXIMUM SIZE?
1643         BGTRU   NOADJUST                ; YES, CAN'T GO ANY LARGER
1644         ADDL3   PCB$L_GPGCNT(R4),PCB$L_PPGCNT(R4),R1    ; GET CURRENT PHYSICAL SIZE
1645         CMPL    R0,R1                   ; Be sure that pages in use don't exceed WS
1646         BLSSU   WSERR                   ; BRANCH IF WS SMALLER THAN PAGES IN USE
1647         ASHL    #-2,R0,R2               ; Compute 75% of WSSIZE as page threshold
1648         SUBL2   R2,R0
1649         CMPL    R1,R0                   ; If threshold not exceeded,
1650         BLSSU   NOADJUST                ; skip WS adjustment
1651 ADJUST: MOVAB   PHD$B_AWSMODE(R5),R0    ; GET ADDRESS OF AST ACCESS MODE
1652         BSBB    SENDAST                 ; SEND AST TO PROCESS
1653
1654 ADJWS:  .WORD   0                       ;
1655         MOVL    G^CTL$GL_PHD,R1         ; GET PHD ADDRESS SO
1656         CLRB    PHD$B_AWSMODE(R1)       ;  ACCESS MODE FLAG CAN BE RESET
1657         $ADJWSL_S       4(AP)           ; ADJUST BY PARAMETER IN AST ARGLIST
1658         RET                             ; AND RETURN
1659
1660 WSERR:  BUG_CHECK WSSIZEERR,FATAL       ; WORKING SET SIZE CALC IN ERROR
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 37
X-32U5 SENDAST - Send AST to process 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;1 (15

 1662          .SBTTL  SENDAST - Send AST to process
 1663 ;++
 1664 ; FUNCTIONAL DESCRIPTION: SENDAST IS CALLED BY G^SCH$QEND TO SEND ASTS TO THE
 1665 ;        PROCESS THAT INVOKE FUNCTIONS UNAVAILABLE TO THE ENVIRONMENT OF G^SCH$QEND.
 1666 ;        THESE INCLUDE ADJUSTING THE WORKING SET AND EXITING.
 1667 ;
 1668 ; INPUT PARAMETERS:
 1669 ;
 1670 ;        R0      - ADDRESS OF ACCESS MODE FOR AST
 1671 ;                    (NEGATIVE CONTENTS PREVENT SENDING AST)
 1672 ;        R3      - AST PARAMETER
 1673 ;        R4      - PCB ADDRESS
 1674 ;        (SP)    - AST ADDRESS
 1675 ;        4(SP)   - RETURN ADDRESS FOR THIS SUBROUTINE
 1676 ;
 1677 ; ENVIRONMENT:
 1678 ;        IPL = IPL$_SYNCH, SCHED DATABASE LOCKED
 1679 ;--
 1680 SENDAST:                                       ;
 1681          PUSHL   R0                            ; SAVE ADDRESS OF ACCESS MODE
 1682          PUSHL   R3                            ; AND AST PARAMETER
 1683          TSTB    (R0)                          ; CHECK VALUE OF ACCESS MODE
 1684          BLSS    10$                           ; DO NOT QUEUE AST IF NEGATIVE
 1685          BBS     #PCB$V_DELPEN,PCB$L_STS(R4),10$ ;  NOR IF MARKED FOR DELETE
 1686          MOVZWL  #ACB$C_LENGTH,R1              ; SET SIZE REQUIRED
 1687          JSB     G^EXE$ALONONPAGED                     ; ALLOCATE A BLOCK
 1688          BLBC    R0,10$                        ; NONE, TRY LATER
 1689          MOVB    #DYN$C_ACB,ACB$B_TYPE(R2); SET TYPE OF STRUCTURE
 1690          MOVW    R1,ACB$W_SIZE(R2)             ; AND SIZE OF STRUCTURE
 1691          MOVL    (SP)+,ACB$L_ASTPRM(R2)        ; AND AST PARAMETER VALUE
 1692          MOVB    @(SP),ACB$B_RMOD(R2)          ; SET ACCESS MODE FOR AST
 1693          DECB    @(SP)+                        ; INDICATE SUCCESS FOR THIS ACCESS MODE
 1694          MOVL    (SP)+,ACB$L_AST(R2)           ; SET AST ADDRESS
 1695          MOVL    PCB$L_PID(R4),ACB$L_PID(R2)     ; SET PID FOR AST
 1696          PUSHL   R5                            ; SAVE REGS FOR QAST
 1697          PUSHL   R4                            ; ...
 1698          MOVL    R2,R5                         ; SET ADDRESS OF ACB
 1699          CLRL    R2                            ; NULL PRIORITY INCREMENT
 1700          JSB     G^SCH$QAST                    ; QUEUE AST FOR PROCESS
 1701          MOVQ    (SP)+,R4                      ; RESTORE PCB,PHD ADDRESSES
 1702          RSB                                   ; EXIT
 1703
 1704 ;  Error path if nonpaged pool allocation fails or if AST access mode is
 1705 ;  negative, indicating either an AST in progress (for automatic working
 1706 ;  set adjustment) or all access modes are done (for CPU time limit expiration)
 1707
 1708 10$:     ADDL    #12,SP                        ; CLEAN PARAMETERS FROM STACK
 1709          RSB                                   ; AND EXIT
 1710
```

```
1712
1713          .SBTTL  SCH$WAKE - WAKE PROCESS INTERNAL
1714 ;++
1715 ; FUNCTIONAL DESCRIPTION:
1716 ;       SCH$WAKE WAKES THE PROCESS SPECIFIED BY THE PID SUPPLIED.
1717 ;
1718 ; CALLING SEQUENCE:
1719 ;       BSB/JSB G^SCH$WAKE
1720 ;
1721 ; INPUT PARAMETERS:
1722 ;       R1 - PID OF PROCESS TO WAKE
1723 ;
1724 ; OUTPUT PARAMETERS:
1725 ;       R0 - COMPLETION STATUS CODE
1726 ;       R4 - PCB ADDRESS OF PROCESS AWAKENED
1727 ;
1728 ; COMPLETION CODES:
1729 ;       SS$_NORMAL  -  NORMAL SUCCESSFUL COMPLETION STATUS
1730 ;       SS$_NONEXPR -  NONEXISTENT PROCESS (INVALID PID)
1731 ;
1732 ; ENVIRONMENT:
1733 ;       IPL = IPL$_SYNCH, SCHED DATABASE LOCKED
1734 ;
1735 ;--
1736          UNIVERSAL_SYMBOL        SCH$WAKE
1737 ;SCH$WAKE::                              ; WAKE PROCESS INTERNAL
1738          MOVZWL  R1,R4                   ; GET PROCESS INDEX (PIX)
1739          MOVL    @L^SCH$GL_PCBVEC[R4],R4 ; LOOK UP PCB ADDRESS
1740          CMPL    R1,PCB$L_PID(R4)        ; VERIFY PID
1741          BNEQ    30$                     ; REPORT ERROR
1742          BBSS    #PCB$V_WAKEPEN,PCB$L_STS(R4),10$   ; SET WAKE PENDING
1743 10$:                                     ;
1744          MOVZBL  #PRI$_RESAVL,R2         ; SET PRIORITY INCREMENT CLASS
1745          RPTEVT  WAKE                    ; REPORT WAKE EVENT
1746          MOVZWL  #SS$_NORMAL,R0          ; SET SUCCESS CODE
1747 20$:     RSB                             ; RETURN
1748
1749 30$:     MOVZWL  #SS$_NONEXPR,R0         ; SET NONEXISTENT PROCESS STATUS
1750          RSB                             ;
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 39
X-32U5 SCH$SWPWAKE - WAKE SWAPPER PROCESS 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RSE.MAR;

 1752          .SBTTL  SCH$SWPWAKE - WAKE SWAPPER PROCESS
 1753 ;++
 1754 ; FUNCTIONAL DESCRIPTION:
 1755 ;       SCH$SWPWAKE AWAKENS THE SWAPPER PROCESS TO PERFORM SOME OPERATION.
 1756 ;
 1757 ; CALLING SEQUENCE:
 1758 ;       BSB/JSB G^SCH$SWPWAKE
 1759 ;
 1760 ; INPUT PARAMETERS:
 1761 ;       NONE
 1762 ;
 1763 ; OUTPUT PARAMETERS:
 1764 ;       R0-R4 PRESERVED
 1765 ;
 1766 ; SIDE EFFECTS:
 1767 ;       A WAKE EVENT IS GENERATED FOR THE SWAPPER PROCESS WHICH CAN
 1768 ;       CAUSE THE PROCESSOR TO BE RESCHEDULED.
 1769 ;
 1770 ; ENVIRONMENT:
 1771 ;       THE SCHED SPINLOCK MUST BE HELD ON ENTRY.
 1772 ;
 1773 ;--
 1774
 1775          UNIVERSAL_SYMBOL        SCH$SWPWAKE
 1776 ;SCH$SWPWAKE::                              ; WAKE SWAPPER PROCESS
 1777          TSTL    G^SCH$GL_COMOQS            ; ANY INSWAP CANDIDATES?
 1778          BNEQ    10$                        ; YES, MUST WAKE SWAPPER THEN
 1779          CMPL    G^SCH$GL_MFYCNT,G^SCH$GL_MFYLIM ; ARE THERE MODIFIED PAGES TO WRITE?
 1780          BGEQ    10$                        ; YES, MUST WAKE SWAPPER THEN
 1781          CMPL    G^SCH$GL_FREECNT,G^SCH$GL_FREELIM ; DO WE NEED FREE PAGES?
 1782          BLSS    10$                        ; YES, MUST WAKE SWAPPER THEN
 1783          TSTW    G^SCH$GW_DELPHDCT          ; ARE THERE DELETED HEADERS TO PURGE?
 1784          BNEQ    10$                        ; YES, MUST WAKE SWAPPER THEN
 1785          TSTL    G^EXE$GL_PFATIM            ; WAS THERE A POWER FAIL RECOVERY?
 1786          BEQL    20$                        ; BR IF NONE
 1787 10$:     BITB    S^#<<1@SCH$V_SIP>!<1@SCH$V_MPW>>,G^SCH$GB_SIP ; SWAPPER ALREADY BUSY
 1788          BNEQ    20$                        ; BR IF YES
 1789          PUSHR   #^M<R0,R1,R2,R3,R4>        ; SAVE R1-R4
 1790          MOVL    G^SCH$GL_SWPPID,R1         ; GET PID OF SWAPPER
 1791          BSBB    SCH$WAKE                   ; AND AWAKEN IT
 1792          POPR    #^M<R0,R1,R2,R3,R4>        ; RESTORE R1-R4
 1793 20$:     RSB                                ; AND RETURN TO CALLER
 1794
```

```
1796          .SBTTL  SCH$CHANGE_CUR_PRIORITY  - Modify priority of running process
1797 ;++
1798 ; FUNCTIONAL DESCRIPTION:
1799 ;       SCH$CHANGE_CUR_PRIORITY is used when the priority of a running process
1800 ;       is to be changed.  It modifies  PCB$B_PRI  and other scheduler data
1801 ;       structures.   No other PCB fileds are modified.  Preemption may take
1802 ;       place.
1803 ;
1804 ; CALLING SEQUENCE:
1805 ;       BSB/JSB SCH$CHANGE_CUR_PRIORITY
1806 ;
1807 ; INPUT PARAMETERS:
1808 ;       R0 - new priority
1809 ;       R1 - CPU database address
1810 ;       R4 - PCB  address
1811 ;
1812 ; OUTPUT PARAMETERS:
1813 ;       R0-R4 PRESERVED
1814 ;
1815 ;
1816 ; ENVIRONMENT:
1817 ;       The SCHED spinlock must be held on entry.
1818 ;
1819 ;--
1820
1821          UNIVERSAL_SYMBOL SCH$CHANGE_CUR_PRIORITY
1822
1823          PUSHL   R2
1824          MOVZBL  PCB$B_PRI(R4),R2          ; Current priority - internal form
1825          BICL    CPU$L_CPUID_MASK(R1),-    ; Get mask for current CPU ID
1826                  W^SCH$AL_CPU_PRIORITY[R2] ; Clear CPU bit
1827          BNEQ    10$                       ; Anyone else at this priority?
1828          SUBL3   R2,#31,R2                 ; Get priority in external format
1829          BBCC    R2,-                      ; No one else
1830                  G^SCH$GL_ACTIVE_PRIORITY,10$
1831 10$:    MOVB    R0,CPU$B_CUR_PRI(R1)      ; New priority for this CPU
1832          MOVB    R0,PCB$B_PRI(R4)
1833          BISL    CPU$L_CPUID_MASK(R1),-    ; Get mask for current CPU ID
1834                  W^SCH$AL_CPU_PRIORITY[R0] ; Set CPU bit
1835          SUBL3   R0,#31,R2                 ; Get priority in external format
1836          BBSS    R2,-
1837                  G^SCH$GL_ACTIVE_PRIORITY,20$
1838 20$:    FFS     #0,#32,G^SCH$GL_COMQS,R2 ; Get highest priority COM process
1839          CMPB    R0,R2
1840          BLEQ    30$                       ; Don't preempt
1841          FIND_CPU_DATA   R2                ; Address for this CPU
1842          CMPL    R2,R1                     ; Is process on this CPU
1843          BNEQ    40$                       ; No
1844          SOFTINT #IPL$_RESCHED
1845 30$:    POPL    R2
1846          RSB
1847
1848 40$:    IPINT_CPU  RESCHED,CPU$L_PHY_CPUID(R1) ; Reschedule the correct CPU
1849          BRB     30$
1850
```

```
RSE - REPORT SYSTEM EVENT 10-MAY-1989 16:47:03 VAX MACRO V5.0-8 Page 41
X-32U5 SCH$CUR_TO_COM - Make the current proces 19-SEP-1988 08:23:26 _$254$DUA55:[SYS.SRC]RS

 1852          .SBTTL  SCH$CUR_TO_COM - Make the current process COM
 1853 ;++
 1854 ; FUNCTIONAL DESCRIPTION:
 1855 ;       SCH$CUR_TO_COM is used to make the current process COM.  It must be
 1856 ;       from the context of the affected process.
 1857 ;
 1858 ; CALLING SEQUENCE:
 1859 ;       BSB/JSB SCH$CUR_TO_COM
 1860 ;
 1861 ; INPUT PARAMETERS:
 1862 ;       R4 - PCB  address
 1863 ;       PC/PSL on stack for a SVPCTX
 1864 ;
 1865 ;       R0  destroyed
 1866 ;
 1867 ; ENVIRONMENT:
 1868 ;       The SCHED spinlock must be held on entry.
 1869 ;
 1870 ;--
 1871
 1872          UNIVERSAL_SYMBOL SCH$CUR_TO_COM
 1873          MOVL    (SP)+,R0                ; return address
 1874          SVPCTX
 1875          PUSHL   R0                      ; get return address on correct stack
 1876          MOVQ    R2,-(SP)
 1877          FIND_CPU_DATA   R3
 1878          MOVZBL  PCB$B_PRI(R4),R0        ; pick up process' current priority
 1879          BICL    CPU$L_CPUID_MASK(R3),-  ; get mask for current CPU ID
 1880                  W^SCH$AL_CPU_PRIORITY[R0] ; clear the CPU bit for this priority
 1881          BNEQ    10$                     ; anyone else at this priority?
 1882          SUBL3   R0,#31,R2               ; get priority in external format
 1883          BBCC    R2,-                    ; no one else at this priority
 1884                  G^SCH$GL_ACTIVE_PRIORITY,10$
 1885 10$:     BBSS    R0,G^SCH$GL_COMQS,20$   ; set summary bit
 1886 20$:     MOVW    #SCH$C_COM,-            ; set state to computable resident
 1887                  PCB$W_STATE(R4)
 1888          MOVAQ   G^SCH$AQ_COMT[R0],R3    ; pick up address of tail of scheduling queu
 1889          INSQUE  (R4),@(R3)+             ; and insert PCB
 1890          CLRL    G^SCH$GL_IDLE_CPUS      ; Tell everyone else
 1891          MOVQ    (SP)+,R2
 1892          RSB
 1893
 1894
 1895          .END
```

3 SCHED.LIS

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 0
Table of contents
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 1
X-20U6 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SCHED.MAR;1 (1)

    1          .TITLE  SCHED RESCHEDULING INTERRUPT HANDLER
    2          .IDENT  'X-20U6'
    3  ;*****************************************************************************
    4  ;*                                                                           *
    5  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
    6  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
    7  ;*  ALL RIGHTS RESERVED.                                                     *
    8  ;*                                                                           *
    9  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
   10  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
   11  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
   12  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
   13  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
   14  ;*  TRANSFERRED.                                                             *
   15  ;*                                                                           *
   16  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
   17  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
   18  ;*  CORPORATION.                                                             *
   19  ;*                                                                           *
   20  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
   21  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
   22  ;*                                                                           *
   23  ;*                                                                           *
   24  ;*****************************************************************************
   25  ;
   26  ; MODIFIED BY:
   27  ;
   28  ;       X-20U6  WMC00U6         Wayne Cardoza           12-Apr-1988
   29  ;               Fix typo in REMOVE_AFFINITY.
   30  ;
   31  ;       X-20U5  EMB0360         Ellen M. Batbouta       01-Nov-1988
   32  ;               While attempting to acquire the SCHED spinlock in SCH$IDLE
   33  ;               set a flag in the per-CPU database to indicate this.  The
   34  ;               purpose of the flag is to help with clock tick accounting.
   35  ;
   36  ;       X-20U4  WMC00U4         Wayne Cardoza           13-Oct-1988
   37  ;               Add CPU-specific idle loop support.
   38  ;
   39  ;       X-20U3  WMC00U3         Wayne Cardoza           28-Sep-1988
   40  ;               Finalize preemption masks.
   41  ;
   42  ;       X-20U2  WMC00U2         Wayne Cardoza           19-Sep-1988
   43  ;               Add support for callback when breaking affinity.
   44  ;
   45  ;       X-20    WMC0020         Wayne Cardoza           12-Sep-1988
   46  ;               Add implicit affinity support.
   47  ;
   48  ;       X-19    WMC0019         Wayne Cardoza           02-Sep-1988
   49  ;               Fix a truncation error.
   50  ;
   51  ;       X-18    WMC0018         Wayne Cardoza           17-Aug-1988
   52  ;               Continue the affinity work.
   53  ;               Add current affinity calculation.
   54  ;
   55  ;       X-17    WMC0017         Wayne Cardoza           12-Aug-1988
   56  ;               Add routines for capability management.
   57  ;
```

```
;       X-16    WMC0016         Wayne Cardoza           30-Jun-1988
;               Add data  cells for capability enhancements.
;               Keep track of all current priorities.
;               Use sequence number to get out of idle loop.
;
;       X-15    MSH0326         Michael S. Harvey       20-Oct-1987
;               Eliminate extraneous MFPR on high-frequency code path,
;               in the FIND_CPU_DATA macro.
;
;       X-14    SSA0004         Stan Amway              21-Sep-1987
;               Check for lost cluster quorum.
;
;       X-13    SSA0003         Stan Amway               8-Sep-1987
;               Maintain idle CPU mask for SMP scheduling support.
;
;       X-12    RNG5012         Rod Gamache             3-Mar-1987
;               Make SCB dispatch directly to SCH$RECHED in this module.
;
;       X-11    RNG5011         Rod Gamache             27-Feb-1987
;               Remove MPH$xxx ASMP multiprocessing hooks.
;
;       X-10    SF04001         Stephen Fiorelli        05-Feb-1987
;               SCH$GL_NULLPCB changed to SCH$AR_NULLPCB.
;
;       X-9     SSA0002         Stan Amway              21-Jan-1987
;               Comment out test and bugcheck for real-time process
;               current priority not equal to base priority. Examination
;               of scheduling related code and testing indicates that
;               the check is not required.
;
;       X-8     WCT0015         Ward C. Travis           6-Jan-1987
;               Update  outdated  SMPLOCK,  SMPUNLOCK uses  to  LOCK,
;               UNLOCK for SMP.
;
;       X-7,6   SJF             Stu Farnham             8-Dec-1986
;               Skip processes prevented from running on this CPU
;               by affinity. Scan instead for the next eligible process.
;               Change reference to g^SCH$GL_CTLFLAGS to W^SCH$$GL_CTLFLAGS.
;
;       X-5     RNG0005         Rod Gamache             28-Oct-1986
;               Change IPL$_SCHED to IPL$_RESCHED.
;
;       X-4     SSA0001         Stan Amway              26-Sep-1986
;               Add check for process priority decrement, and bugcheck
;               for real-time process current priority not equal to
;               base priority. The latter check will be removed for
;               the SDC release of V5.0.
;
;       X-2H2   SJF             Stu Farnham             3-June-1986
;               Resolve merge conflicts
;
;       X-1F2   MSH0242         Michael S. Harvey       8-May-1986
;               Replace logical CPU ID with physical CPU ID.
;
;       X-1A6   RNG4006         Rod Gamache             5-Feb-1986
;               Save logical CPU index in PCB.
;
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 3
X-20U6 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SCHED.MAR;1 (1)

;       X-1A5   RNG4005         Rod Gamache             21-Jan-1986
;               Don't run NULL process, instead run idle loop on interrupt
;               stack at IPL 3.
;
;       X-1A4   SJF             Stu Farnham             27-Dec-1985
;               Change refs to CPU$B_PRI to CPU$B_CUR_PRI
;
;       X-1A3   SJF             Stu Farnham             27-Dec-1985
;               Add per-cpu structure refs
;
;       X-1A2   RNG4002         Rod Gamache             25-Jul-1985
;               Change SETIPLs to SMPLOCKS/SMPUNLOCKS and change REI to
;               an REIMAC macro.
;
;       V03-006 KPL0001         Peter Lieberwirth       15-Jan-1984
;               Fix some broken work displacements, in fact, fix them all.
;
;       V03-005 TMK0001         Todd M. Katz            20-Dec-1983
;               Fix some broken word displacements.
;
;       V03-004 WMC0004         Wayne Cardoza           10-Jan-1982
;               Fix some broken word displacements.
;
;       V03-003 JWH0143         Jeffrey W. Horn         30-Nov-1982
;               Back-out JWH0136, code that broke references is now in
;               another PSECT.
;
;       V03-002 JWH0136         Jeffrey W. Horn         24-Nov-1982
;               Fix broken references.
;
;       V03-001 KDM0083         Kathleen D. Morse       31-Mar-1982
;               Add multi-processing hooks for replacement scheduling routines.
;
;
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 4
X-20U6 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SCHED.MAR;1 (1)

  150 ; INCLUDE FILES:
  151
  152         $CPBDEF                         ; CPU capability number definitions
  153         $CPUDEF                         ;PER-CPU DATA BLOCK OFFSETS
  154         $DYNDEF                         ;STRUCTURE TYPE CODE DEFINITIONS
  155         $IPLDEF                         ;INTERRUPT PRIORITY LEVEL DEFINITIONS
  156         $PCBDEF                         ;PCB DEFINITIONS
  157         $PHDDEF                         ;PHD DEFINITIONS
  158         $PRDEF                          ;PROCESSOR REGISTER DEFINITIONS
  159         $RSNDEF                         ; Resource number definitions
  160         $SSDEF                          ;STATUS CODES
  161         $STATEDEF                       ;STATE DEFINITIONS
  162         $WQHDEF                         ;WAIT QUEUES
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 5
X-20U6 Scheduler Data 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SCHED.MAR;1 (2)

  164          .SBTTL  Scheduler Data
  165
  166          DECLARE_PSECT   EXEC$NONPAGED_DATA      ; NONPAGED EXEC
  167
  168 ;
  169 ; Array of priorities preemptable by each priority.
  170 ; Mask assume  comparison  against external priorities, but is indexed by
  171 ; internal priority.
  172 ;
  173 SCH$AL_PREEMPT_MASK::
  174          .LONG   ^B01111111111111111111111111111111
  175          .LONG   ^B00111111111111111111111111111111
  176          .LONG   ^B00011111111111111111111111111111
  177          .LONG   ^B00001111111111111111111111111111
  178          .LONG   ^B00000111111111111111111111111111
  179          .LONG   ^B00000011111111111111111111111111
  180          .LONG   ^B00000001111111111111111111111111
  181          .LONG   ^B00000000111111111111111111111111
  182          .LONG   ^B00000000011111111111111111111111
  183          .LONG   ^B00000000001111111111111111111111
  184          .LONG   ^B00000000000111111111111111111111
  185          .LONG   ^B00000000000011111111111111111111
  186          .LONG   ^B00000000000001111111111111111111
  187          .LONG   ^B00000000000000111111111111111111
  188          .LONG   ^B00000000000000011111111111111111
  189          .LONG   ^B00000000000000001111111111111111
  190          .LONG   ^B00000000000000000011111111111111
  191          .LONG   ^B00000000000000000001111111111111
  192          .LONG   ^B00000000000000000000111111111111
  193          .LONG   ^B00000000000000000000011111111111
  194          .LONG   ^B00000000000000000000001111111111
  195          .LONG   ^B00000000000000000000000111111111
  196          .LONG   ^B00000000000000000000000011111111
  197          .LONG   ^B00000000000000000000000001111111
  198          .LONG   ^B00000000000000000000000000111111
  199          .LONG   ^B00000000000000000000000000011111
  200          .LONG   ^B00000000000000000000000000001111
  201          .LONG   ^B00000000000000000000000000000111
  202          .LONG   ^B00000000000000000000000000000011
  203          .LONG   ^B00000000000000000000000000000001
  204          .LONG   ^B00000000000000000000000000000000
  205          .LONG   ^B00000000000000000000000000000000
  206 ;
  207 ; Array of masks indicating which CPU is at each priority.
  208 ; Indexed by priority.  Bits are CPU IDs.
  209 ;
  210 SCH$AL_CPU_PRIORITY::
  211          .BLKL   32
  212 ;
  213 ; Array of capabilities held by each CPU.  Duplicates CPU database.
  214 ; Indexed by CPU ID.
  215 ;
  216 SCH$AL_CPU_CAP::
  217          .BLKL   32
  218 ;
  219 ; Sequence number for changes to CPU capabilities.
  220 ;
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 6
X-2006 Scheduler Data 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SCHED.MAR;1 (2)

  221 SCH$GL_CAPABILITY_SEQUENCE::
  222          .LONG   1
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 7
X-20U6 Initialization Routine 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SCHED.MAR;1 (3)

  224          .SBTTL  Initialization Routine
  225 ;
  226 ; SCHED_INIT
  227 ;
  228 ;       Initialize the SCB with the RESCHED handler address.
  229 ;
  230 ;
  231          DECLARE_PSECT EXEC$INIT_CODE
  232
  233 INITIALIZATION_ROUTINE -
  234          SCHED_INIT
  235
  236 SCHED_INIT:
  237          MOVL    G^EXE$GL_SCB,R0                 ; SCB address
  238          MOVAB   SCH$RESCHED,^X8C(R0)            ; Fill in RESCHED vector
  239
  240          FIND_CPU_DATA   R0                      ; Get our CPU database
  241          MOVL    CPU$L_PHY_CPUID(R0),R1          ; CPU ID (primary)
  242          MOVL    CPU$L_CAPABILITY(R0),-          ; Primary CPU capabilities
  243                  SCH$AL_CPU_CAP[R1]
  244
  245          MOVL    #SS$_NORMAL,R0
  246          RSB
  247
  248
  249          DECLARE_PSECT   EXEC$NONPAGED_CODE      ; NONPAGED EXEC
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 8
X-2006 SCH$RESCHED RESCHEDULING INTERRUPT HANDL 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SC

                        .SBTTL  SCH$RESCHED RESCHEDULING INTERRUPT HANDLER
                ;++
                ; SCH$RESCHED - RESCHEDULING INTERRUPT HANDLER
                ;
                ; THIS ROUTINE IS ENTERED VIA THE IPL 3 RESCHEDULING INTERRUPT.
                ; THE VECTOR FOR THIS INTERRUPT IS CODED TO CAUSE EXECUTION
                ; ON THE KERNEL STACK.
                ;
                ; ENVIRONMENT:
                ;       IPL=3 MODE=KERNEL IS=0
                ; INPUT:
                ;       00(SP)=PC AT RESCHEDULE INTERRUPT
                ;       04(SP)=PSL AT INTERRUPT.
                ;--
                        .ENABL  LSB

                        ASSUME  PCB$V_QUANTUM_RESCHED EQ 0
                ;15$:    BLBS    PCB$L_STS2(R1),16$      ; BR if not pre-emption reschedule
                ;       INSQUE  (R1),-4(R2)             ;INSERT AT HEAD OF QUEUE
                ;151$:  BRW     30$
                ;
                ;16$:    INSQUE  (R1),@(R2)+             ;INSERT AT TAIL OF QUEUE
                ;       BBSC    #PCB$V_QUANTUM_RESCHED,-
                ;               PCB$L_STS2(R1),151$
                ;       BRW     SCH$IDLE

                        .ALIGN  LONG
                        UNIVERSAL_SYMBOL        SCH$RESCHED
                ;SCH$RESCHED::                           ;RESCHEDULE INTERRUPT HANDLER
                        LOCK    LOCKNAME=SCHED,-        ;LOCK SCHED DATABASE
                                LOCKIPL=#IPL$_SYNCH     ;RAISE TO SYNCH IPL
                        SVPCTX                          ;SAVE CONTEXT OF PROCESS
                        FIND_CPU_DATA   R3,ISTACK=YES   ;GET THIS CPU'S PER-CPU DATABASE
                                                        ; (WE CAN ASSUME WE'RE ON I-STACK)
                        MOVL    CPU$L_CURPCB(R3),R1     ;GET ADDRESS OF CURRENT PCB
                        MOVZBL  CPU$B_CUR_PRI(R3),R0    ;CURRENT PRIORITY
                        BICL    CPU$L_CPUID_MASK(R3),-  ;GET MASK FOR CURRENT CPU ID
                                W^SCH$AL_CPU_PRIORITY[R0] ;CLEAR CPU BIT
                        BNEQ    5$                      ;ANYONE ELSE AT THIS PRIORITY?
                        SUBL3   R0,#31,R2               ;GET PRIORITY IN EXTERNAL FORMAT
                        BBCC    R2,-                    ;NO ONE ELSE
                                G^SCH$GL_ACTIVE_PRIORITY,5$
                5$:     BBSS    R0,G^SCH$GL_COMQS,10$   ;MARK QUEUE NON-EMPTY
                10$:    MOVW    #SCH$C_COM,PCB$W_STATE(R1) ;SET STATE TO RES COMPUTE
                        MOVAQ   G^SCH$AQ_COMT[R0],R2    ; R2 = address of queue header back link
                ;       BBS     S^#SCH$V_PREEMPT_RESUME,-; BR if pre-emptive resume
                ;               W^SCH$$GL_CTLFLAGS,15$  ; scheduling enabled
                        INSQUE  (R1),@(R2)+             ;INSERT AT TAIL OF QUEUE
                        CLRL    G^SCH$GL_IDLE_CPUS      ;TELL EVERYONE ELSE
                        BRW     30$                     ;SKIP ACQUIRING SPINLOCK AGAIN

                SCH$IDLE:                               ;NO ACTIVE, EXECUTABLE PROCESS
                        BISL2   CPU$L_CPUID_MASK(R3),-  ; Show this CPU as idle
                                G^SCH$GL_IDLE_CPUS
                        MOVL    G^SCH$AR_NULLPCB,CPU$L_CURPCB(R3) ;NOTE NULL PCB AS DEFAULT
                        MNEGB   #1,CPU$B_CUR_PRI(R3)    ;SET PRIORITY TO -1 TO SIGNAL IDLE
                        UNLOCK  LOCKNAME=SCHED,-        ;UNLOCK SCHED DATABASE
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 9
X-2006 SCH$RESCHED RESCHEDULING INTERRUPT HANDL 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SC

                        NEWIPL=#IPL$_RESCHED    ;DROP IPL TO RESCHEDULING LEVEL
                MOVL    CPU$L_PHY_CPUID(R3),R1  ;GET OUR CPU ID
60$:            JSB     G^EXE$PROC_IDLE         ;SEE IF WE HAVE  CPU SPECIFIC IDLE LOOP
                BLBS    R0,61$                  ;NO - GO TO STANDARD LOOP
                BBS     R1,G^SCH$GL_IDLE_CPUS,60$ ;LOOP UNTIL WE AREN'T IDLE
                BRB     65$
61$:            BBS     R1,G^SCH$GL_IDLE_CPUS,61$ ;LOOP UNTIL WE AREN'T IDLE
65$:            BISB    #CPU$M_SCHED,CPU$B_FLAGS(R3) ;INDICATE IDLE VYING FOR SCHED
                LOCK    LOCKNAME=SCHED,-        ;LOCK SCHED DATABASE
                        LOCKIPL=#IPL$_SYNCH     ;RAISE TO SYNCH IPL
                BICB    #CPU$M_SCHED,CPU$B_FLAGS(R3) ;INDICATE NO LONGER VYING FOR SCHED
                BRW     30$                     ;GO TRY FOR PROCESS

19$:            BRB     SCH$IDLE

QEMPTY: BUG_CHECK QUEUEMPTY,FATAL               ;SCHEDULING QUEUE EMPTY

;+
; SCH$SCHED - SCHEDULE NEW PROCESS FOR EXECUTION
;
; THIS ROUTINE SELECTS THE HIGHEST PRIORITY EXECUTABLE PROCESS
; AND PLACES IT IN EXECUTION.
;
;-
                .ALIGN  LONG
                UNIVERSAL_SYMBOL        SCH$SCHED
;SCH$SCHED::                                    ;SCHEDULE FOR EXECUTION
                FIND_CPU_DATA   R3,ISTACK=YES   ;GET BASE OF PER-CPU DATA
                                                ; (WE CAN ASSUME I-STACK HERE)
                LOCK    LOCKNAME=SCHED,-        ;LOCK SCHED DATABASE
                        LOCKIPL=#IPL$_SYNCH     ;RAISE TO SYNCH IPL

                MOVZBL  CPU$B_CUR_PRI(R3),R0    ;GET PREVIOUS CPU PRIORITY
                BICL    CPU$L_CPUID_MASK(R3),-  ;GET MASK FOR CURRENT CPU ID
                        W^SCH$AL_CPU_PRIORITY[R0] ;CLEAR CPU BIT
                BNEQ    30$                     ;ANYONE ELSE AT THIS PRIORITY?
                SUBL3   R0,#31,R2               ;GET PRIORITY IN EXTERNAL FORMAT
                BBCC    R2,-                    ;NO ONE ELSE
                        G^SCH$GL_ACTIVE_PRIORITY ,30$
30$:            CLRL    R7                      ; CLEAR IMPLICIT AFFINITY STATE
                FFS     #0,#32,G^SCH$GL_COMQS,R0  ;FIND FIRST FULL STATE
                BEQL    19$                     ;NO EXECUTABLE PROCESS?
35$:            MOVAQ   G^SCH$AQ_COMH[R0],R2    ;COMPUTE QUEUE HEAD ADDRESS
                REMQUE  @(R2)+,R4               ;GET HEAD OF QUEUE
38$:            BVS     QEMPTY                  ;BR IF QUEUE WAS EMPTY (BUG CHECK)
                BNEQ    40$                     ;QUEUE NOT EMPTY
                BBCC    R0,G^SCH$GL_COMQS,40$   ;SET QUEUE EMPTY
40$:                                            ;
                CMPB    #DYN$C_PCB,PCB$B_TYPE(R4)  ;MUST BE A PROCESS CONTROL BLOCK
                BNEQ    QEMPTY                  ;OTHERWISE FATAL ERROR
                BICL3   CPU$L_CAPABILITY(R3),-  ;DO THE CPU AND PROCESS MATCH
                        PCB$L_CAPABILITY(R4),R1
                BNEQ    200$                    ;NO
45$:            MOVL    R4,CPU$L_CURPCB(R3)     ;NOTE CURRENT PCB LOC
                MOVW    #SCH$C_CUR,PCB$W_STATE(R4) ;SET STATE TO CURRENT
                MOVL    CPU$L_PHY_CPUID(R3),PCB$L_CPU_ID(R4) ; SAVE CPU ID IN PCB
                                                ; The following assumes that
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 10
X-2006 SCH$RESCHED RESCHEDULING INTERRUPT HANDL 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SC

                                                ; SCH$V_NO_RESCHED_PRIADJ = 0.
                                                ; (SYSPARAM documents this assumption.)
;       BLBS    W^SCH$$GL_CTLFLAGS,50$  ; BR if priority adjustment being
;                                       ; done at quantum end (or elsewhere)
        CMPB    R0,PCB$B_PRIB(R4)       ;CHECK FOR BASE PRIORITY=CURRENT
                                        ;  - SHOULD NEVER BE GTR
        BGEQ    50$                     ;YES, DONT FLOAT PRIORITY
        INCB    R0                      ;MOVE TOWARD BASE PRIO
        MOVB    R0,PCB$B_PRI(R4)        ; Reflect priority change in PCB
50$:    MOVB    R0,CPU$B_CUR_PRI(R3)    ;SET GLOBAL PRIORITY
        MOVL    CPU$L_CPUID_MASK(R3),R2 ;GET CPU MASK
        BICL2   R2,G^SCH$GL_IDLE_CPUS   ; Show this CPU as not idle
        BISL    R2,W^SCH$AL_CPU_PRIORITY[R0] ;SET CPU BIT
        SUBL3   R0,#31,R2               ;GET PRIORITY IN EXTERNAL FORMAT
        BBSS    R2,G^SCH$GL_ACTIVE_PRIORITY,51$ ;PRIORITY NOW ACTIVE
51$:    MTPR    PCB$L_PHYPCB(R4),#PR$_PCBB      ;SET PCB BASE PHYS ADDR
        LDPCTX                          ;RESTORE CONTEXT
        UNLOCK  LOCKNAME=SCHED  ;UNLOCK SCHED DATABASE/NO IPL CHANGE
        REIMAC                          ;NORMAL RETURN

;BADRTPRI:
;       BUG_CHECK BADRTPRI,FATAL

; Make assorted checks to determine the type of capabilty mismatch
;
200$:   BICL    #CPB$M_IMPLICIT_AFFINITY,R1
        BNEQ    300$                    ; Its more than implicit affinity
;
; Deal with implicit affinity
;
        CMPL    PCB$L_AFFINITY(R4),-    ; Does it match this CPU
                CPU$L_PHY_CPUID(R3)
        BNEQ    210$                    ; No
        ADDB3   G^SCH$GL_AFFINITY_SKIP,-; Reinit the skip count
                #1,PCB$B_AFFINITY_SKIP(R4)
205$:   BRB     45$                     ; Go run it

210$:   DECB    PCB$B_AFFINITY_SKIP(R4) ; Update skip count
        BNEQ    220$
        BRW     700$                    ; Go break affinity
220$:   TSTL    R7                      ; Do we already have a skipped process
        BNEQ    500$                    ; Yes - go skip this one
        MOVL    R4,R7                   ; No - record this PCB
        BRB     500$                    ; Go skip the process

300$:   CMPL    W^SCH$GL_CAPABILITY_SEQUENCE,- ; Do sequence numbers match?
                PCB$L_CAPABILITY_SEQ(R4)
        BEQL    310$                            ; Yes - go ahead
        BSBW    SCH$CALCULATE_AFFINITY          ; Get new CPU mask
310$:   BBS     CPU$L_PHY_CPUID(R3),-           ; Can it run on this CPU
                PCB$L_CURRENT_AFFINITY(R4),205$;   Checks explicit affinity only
        TSTL    PCB$L_CURRENT_AFFINITY(R4)       ; Can it run anywhere
        BNEQ    500$                            ; Yes  - try again later
        BRB     WAIT_FOR_CAP                    ; Go wait for capability change

;
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 11
X-20U6 SCH$RESCHED RESCHEDULING INTERRUPT HANDL 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SC

  422 ; Process removed from queue does not have affinity for this cpu. Replace
  423 ; process in queue and find a new candidate if one exists.
  424 ;
  425 500$:   INSQUE  (R4),@4(R4)              ; Replace process in queue
  426         BBSS    R0,G^SCH$GL_COMQS,520$  ; Mark queue non-empty
  427 520$:   CMPL    R4,(R2)                 ; End of queue
  428         BEQL    550$                    ; Yes...
  429         REMQUE  @(R4),R4                ; Get the next one
  430         BRW     40$                     ; Go try again
  431 ;
  432 ; All processes in current queue lack affinity for this CPU. Select next
  433 ; queue unless we have implicit affinity process to run.
  434 ;
  435 550$:   INCL    R0                      ; Start scan at this position plus 1
  436         SUBL3   R0,#32,R5               ; Remaining bits to scan
  437         FFS     R0,R5,G^SCH$GL_COMQS,R0 ; Look for next non-empty queue
  438         BEQL    580$                    ; No more queues
  439         TSTL    R7                      ; Is there implicit affinity candidate
  440         BNEQ    560$                    ; Yes
  441 555$:   BRW     35$                     ; Process next queue
  442 560$:   MOVZBL  PCB$B_PRI(R7),R5        ; Get its internal priority
  443         MOVL    SCH$AL_PREEMPT_MASK[R5],R6 ; Get its preemption mask
  444         SUBL3   R0,#31,R2               ; External priority of next queue
  445         BBC     R2,R6,555$              ; Continue if process would not preempt
  446         MOVL    R5,R0                   ; Get priority in right register
  447 565$:   REMQUE  (R7),R4                 ; Remove it from queue
  448         BNEQ    570$                    ; Queue not empty
  449         BBCC    R0,G^SCH$GL_COMQS,560$  ; Set queue empty
  450 570$:   BRB     700$                    ; Go break affinity
  451
  452 580$:   TSTL    R7                      ; Do we have implicit affinity
  453         BNEQ    590$                    ; Yes
  454         BRW     SCH$IDLE
  455 590$:   MOVZBL  PCB$B_PRI(R7),R0        ; Get its internal priority
  456         BRB     565$                    ; Go run it
  457
  458
  459 ;
  460 ; No CPU in the active set has the necessary capabilities to run this
  461 ; process.  Put it in a wait state until something happens.
  462 ;
  463 WAIT_FOR_CAP:
  464         MOVL    #RSN$_CPUCAP,R5         ; Resource to wait for
  465         BBSSI   R5,G^SCH$GL_RESMASK,600$; Indicate someone is on queue
  466 600$:   MOVL    R5,PCB$L_EFWM(R4)       ; Reason for wait
  467         MOVL    (R4),R5                 ; Save address of next PCB
  468         INSQUE  (R4),G^SCH$GQ_MWAIT     ; Put it on wait queue
  469         INCW    G^SCH$GQ_MWAIT+WQH$W_WQCNT  ; Count it
  470         MOVW    #SCH$C_MWAIT,-          ; Now in wait state
  471                 PCB$W_STATE(R4)
  472         MOVL    G^EXE$GL_ABSTIM_TICS,-  ; Save time wait started
  473                 PCB$L_WAITIME(R4)
  474         SUBL3   #4,R2,R6                ; Get  address of queue header
  475         CMPL    R6,R5                   ; Was it the last element
  476         BEQL    550$                    ; Yes - go try the next queue
  477         REMQUE  (R5),R4                 ; Get the next PCB
  478         BRW     38$                     ; continue
```

```
  479
  480 ;
  481 ; Break implicit affinity for this process
  482 ;
  483 700$:   ADDB3   G^SCH$GL_AFFINITY_SKIP,-; Reinit the skip count
  484                 #1,PCB$B_AFFINITY_SKIP(R4)
  485         MOVL    PCB$L_AFFINITY_CALLBACK(R4),R1 ; Was callback requested
  486         BEQL    710$                    ; No
  487         PUSHL   R0                      ; Save register
  488         PUSHL   PCB$L_AFFINITY(R4)      ; Current CPU
  489         PUSHL   R4                      ; PCB
  490         CALLS   #2,@PCB$L_AFFINITY_CALLBACK(R4) ; Call it
  491         POPL    R0                      ; Restore register
  492         CLRL    PCB$L_AFFINITY_CALLBACK(R4) ; Clear callback
  493 710$:   MOVL    CPU$L_PHY_CPUID(R3),-   ; Init the current CPU
  494                 PCB$L_AFFINITY(R4)
  495         BRW     45$                     ; Go run it
  496
  497         .DSABL  LSB
```

```
  499          .SBTTL  SCH$REQUIRE_CAPABILITY - Add capability to required list
  500 ;++
  501 ; FUNCTIONAL DESCRIPTION:
  502 ;       SCH$REQUIRE_CAPABILITY is called to add the requirement for a set of
  503 ;       capabilities to a process.
  504 ;
  505 ; CALLING SEQUENCE:
  506 ;       CALLS  #5,SCH$REQUIRE_CAPABILITY
  507 ;
  508 ; INPUT PARAMETERS:
  509 ;       arg1 =>  PCB of the affected process
  510 ;
  511 ;       arg2 =   Capability mask
  512 ;
  513 ;       arg3 =   CPU number if explicit affinity was specified in arg2
  514 ;
  515 ;       arg4 =   flags word
  516 ;
  517 ;       arg5 =>  longword to receive previous capability mask (optional)
  518 ;
  519 ;
  520 ;       FLAGS WORD:
  521 ;
  522 ;       CPB$M_FLAG_CHECK_CPU
  523 ;               Return an error if the required capabilities will not
  524 ;               allow the process to execute on any CPU.
  525 ;       CPB$M_FLAG_PERMANENT
  526 ;               Affect the process permanent required capabilities.
  527 ;
  528 ;
  529 ; ENVIRONMENT:
  530 ;       The environment must allow the SCHED lock to be obtained.
  531 ;
  532 ;--
  533
  534          PCB     = 4
  535          MASK    = 8
  536          CPU     = 12
  537          FLAGS   = 16
  538          OLD_MASK= 20
  539
  540          UNIVERSAL_ENTRY SCH$REQUIRE_CAPABILITY,<^M<R2,R3,R4>>
  541
  542          LOCK    LOCKNAME=SCHED,-
  543                  SAVIPL=-(SP),-
  544                  LOCKIPL=#IPL$_SYNCH
  545          MOVL    PCB(AP),R4              ; PCB of affected process
  546          MOVL    PCB$L_CAPABILITY(R4),R2 ; Get old capability mask
  547          MOVL    OLD_MASK(AP),R0         ; Is old mask requested
  548          BEQL    10$                     ; No
  549          MOVL    R2,(R0)                 ; Return it
  550 10$:     BISL    MASK(AP),R2             ; New capability requirements
  551          BICL    #CPB$M_IMPLICIT_AFFINITY,R2 ; Ignore implicit affinity
  552          BBC     #CPB$V_FLAG_CHECK_CPU,- ; Should we ensure process is runnable?
  553                  FLAGS(AP),100$
  554          BBCC    #CPB$V_EXPLICIT_AFFINITY,- ; Is explicit affinity requested
  555                  R2,40$                     ; in the new combined mask
```

```
556          MOVL    PCB$L_AFFINITY(R4),R0      ; Get old affinity
557          BBC     #CPB$V_EXPLICIT_AFFINITY,- ; Is explicit affinity requested
558                  MASK(AP),20$               ; by this call
559          MOVL    CPU(AP),R0                 ; New explicit affinity CPU ID
560 20$:     BICL3   W^SCH$AL_CPU_CAP[R0],R2,R0 ; Can it run here
561          BEQL    100$                       ; Yes
562          BRB     60$                        ; No
563 ;
564 ;        The following code does not lock the active set.  This is OK, since
565 ;        we don't care about new CPUs being added.  We just say the process is
566 ;        not runnable and that was true microseconds ago.  If a CPU is
567 ;        subtracted, it could just as easily have happened microseconds in the
568 ;        future.
569 ;
570 40$:     CLRL    R0                         ; Start with CPU 0
571 50$:     SUBL3   R0,#32,R1                  ; Number of bits to check
572          FFS     R0,R1,G^SMP$GL_ACTIVE_CPUS,R0 ; Get next CPU ID
573          BEQL    60$                        ; No more
574          BICL3   W^SCH$AL_CPU_CAP[R0],R2,R1 ; Can we run here
575          BEQL    100$                       ; Yes
576          INCL    R0                         ; Next
577          BRB     50$                        ; Try more
578 60$:     MOVL    #SS$_BADPARAM,R0           ; Return error
579          BRW     500$
580
581 100$:    ASSUME  CPB$V_EXPLICIT_AFFINITY EQ 31
582          TSTL    MASK(AP)                   ; Check for explicit affinity
583          BGEQ    120$                       ; Not requested
584          TSTL    PCB$L_CAPABILITY(R4)
585          BGEQ    110$                       ; Not currently in effect
586          MOVL    PCB$L_AFFINITY(R4),R0      ; Old explicit affinity
587          MOVL    G^SMP$GL_CPU_DATA[R0],R0   ; Get the CPU database
588          DECW    CPU$W_HARDAFF(R0)          ; Account for the old affinity
589 110$:     MOVL    CPU(AP),R1                 ; Get new affinity
590          MOVL    G^SMP$GL_CPU_DATA[R1],R0   ; Get the CPU database
591          INCW    CPU$W_HARDAFF(R0)          ; Account for the new affinity
592          MOVL    R1,PCB$L_AFFINITY(R4)      ; Record the new affinity
593          BBC     #CPB$V_FLAG_PERMANENT,-    ; Change the permanent capabilities?
594                  FLAGS(AP),120$
595          MOVL    R1,PCB$L_PERMANENT_CPU_AFFINITY(R4); Record the new affinity
596 120$:     BISL    MASK(AP),PCB$L_CAPABILITY(R4) ; New mask
597          BBC     #CPB$V_FLAG_PERMANENT,-    ; Change the permanent capabilities?
598                  FLAGS(AP),125$
599          BISL    MASK(AP),PCB$L_PERMANENT_CAPABILITY(R4) ; New mask
600 125$:     BSBW    SCH$CALCULATE_AFFINITY     ; Calculate new CPU mask
601          CMPW    PCB$W_STATE(R4),#SCH$C_CUR ; Is it currently executing
602          BNEQ    400$                       ; No - all done
603          MOVL    PCB$L_CPU_ID(R4),R0        ; CPU where it is running
604          BBS     R0,PCB$L_CURRENT_AFFINITY(R4),400$ ; Can we still run there?
605 130$:     FIND_CPU_DATA R1                   ; Get our CPU database
606          CMPL    R0,CPU$L_PHY_CPUID(R1)     ; Is it this CPU
607          BNEQ    150$                       ; No - do an expensive RESCHED
608          SOFTINT #IPL$_RESCHED
609          BRB     400$
610 150$:     IPINT_CPU RESCHED,R0
611 400$:     MOVL    #SS$_NORMAL,R0
612 500$:     UNLOCK  LOCKNAME=SCHED,-
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 15
X-2006 SCH$REQUIRE_CAPABILITY - Add capability 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SCH
```

```
  613                     PRESERVE=YES,-
  614                     NEWIPL=(SP)+,-
  615                     CONDITION=RESTORE
  616             RET
```

```
618          .SBTTL  SCH$RELEASE_CAPABILITY - Remove capability from required list
619 ;++
620 ; FUNCTIONAL DESCRIPTION:
621 ;       SCH$RELEASE_CAPABILITY is called to remove the requirement for a set of
622 ;       capabilities to from a process.
623 ;
624 ; CALLING SEQUENCE:
625 ;       CALLS  #5,SCH$RELEASE_CAPABILITY
626 ;
627 ; INPUT PARAMETERS:
628 ;       arg1 =>  PCB of the affected process
629 ;
630 ;       arg2 =   Capability mask
631 ;
632 ;       arg3 =   CPU number if explicit affinity was specified in arg2
633 ;
634 ;       arg4 =   flags word
635 ;
636 ;       arg5 =>  longword to receive previous capability mask (optional)
637 ;
638 ;
639 ;       FLAGS WORD:
640 ;
641 ;       CPB$M_FLAG_PERMANENT
642 ;               Affect the process permanent required capabilities.
643 ;
644 ;
645 ; ENVIRONMENT:
646 ;       The environment must allow the SCHED lock to be obtained.
647 ;
648 ;--
649
650          PCB     = 4
651          MASK    = 8
652          CPU     = 12
653          FLAGS   = 16
654          OLD_MASK= 20
655
656          UNIVERSAL_ENTRY SCH$RELEASE_CAPABILITY,<^M<R4>>
657
658
659          LOCK    LOCKNAME=SCHED,-
660                  SAVIPL=-(SP),-
661                  LOCKIPL=#IPL$_SYNCH
662          MOVL    PCB(AP),R4              ; PCB of affected process
663          MOVL    OLD_MASK(AP),R0         ; Is old mask requested
664          BEQL    10$                     ; No
665          MOVL    PCB$L_CAPABILITY(R4),(R0)  ; Return it
666 10$:    MOVL    MASK(AP),R1                ; New capability requirements
667          BBC     #CPB$V_EXPLICIT_AFFINITY,- ; Is explicit affinity released
668                  R1,100$                    ; by this call
669          MOVL    CPU(AP),R0                 ; Get the CPU number
670          CMPL    R0,PCB$L_AFFINITY(R4)      ; Does explicit affinity match
671          BNEQ    50$                        ; No
672          CLRL    PCB$L_AFFINITY(R4)         ; Clear it
673          MOVL    G^SMP$GL_CPU_DATA[R0],R0   ; Get the CPU data address
674          DECW    CPU$W_HARDAFF(R0)          ; Account for the loss of affinity
```

```
  675          BRB     100$
  676 50$:     BICL    #CPB$M_EXPLICIT_AFFINITY,R1 ; Clear the bit
  677 100$:    BICL    R1,PCB$L_CAPABILITY(R4)     ; Record the new mask
  678          BBC     #CPB$V_FLAG_PERMANENT,-     ; Are the permanent ones affected?
  679                  FLAGS(AP),150$
  680          MOVL    MASK(AP),R1                 ; New copy of capability requirements
  681          BBC     #CPB$V_EXPLICIT_AFFINITY,-  ; Is explicit affinity released
  682                  R1,140$                     ; by this call
  683          MOVL    CPU(AP),R0                  ; Get the CPU number
  684          CMPL    R0,PCB$L_PERMANENT_CPU_AFFINITY(R4) ; Does it match
  685          BNEQ    130$                        ; No
  686          CLRL    PCB$L_PERMANENT_CPU_AFFINITY(R4)    ; Clear it
  687          BRB     140$
  688 130$:    BICL    #CPB$M_EXPLICIT_AFFINITY,R1         ; Clear the bit
  689 140$:    BICL    R1,PCB$L_PERMANENT_CAPABILITY(R4)   ; Record the new mask
  690 150$:    BSBW    SCH$CALCULATE_AFFINITY      ; Calculate new set of CPUs
  691          MOVL    #SS$_NORMAL,R0
  692          UNLOCK  LOCKNAME=SCHED,-
  693                  PRESERVE=YES,-
  694                  NEWIPL=(SP)+,-
  695                  CONDITION=RESTORE
  696          RET
  697
```

```
  699          .SBTTL  SCH$ADD_CPU_CAP, SCH$REMOVE_CPU_CAP - add or remove CPU capability
  700 ;++
  701 ; FUNCTIONAL DESCRIPTION:
  702 ;        SCH$ADD_CPU_CAP  and  SCH$REMOVE_CPU_CAP are called to add or remove a
  703 ;        capabilities from the list of capabilities for a CPU
  704 ;
  705 ;        If all CPUs are specified (-1 for CPU number), the  default CPU
  706 ;        capability mask is also affected.
  707 ;
  708 ; CALLING SEQUENCE:
  709 ;        CALLS  #3,SCH$ADD_CPU_CAP
  710 ;        CALLS  #3,SCH$REMOVE_CPU_CAP
  711 ;
  712 ; INPUT PARAMETERS:
  713 ;        arg1 =>  CPU NUMBER (-1 indicates all cpus)
  714 ;
  715 ;        arg2 =   Capability mask
  716 ;
  717 ;        arg3 =>  longword to receive previous capability mask (optional)
  718 ;                 if all CPUs requested, old default will be returned
  719 ;
  720 ;
  721 ;        FLAGS WORD:
  722 ;
  723 ;        Not applicable at this time
  724 ;
  725 ;
  726 ; ENVIRONMENT:
  727 ;        The SCHED lock must  be held when the routine is called.
  728 ;
  729 ;--
  730
  731          CPU      = 4
  732          MASK     = 8
  733          OLD_MASK= 12
  734
  735          .ENABL  LSB
  736
  737          UNIVERSAL_ENTRY SCH$ADD_CPU_CAP,<^M<R2,R3,R4,R5>>
  738
  739          CLRL     R5                          ; Indicate ADD
  740          BRB      1$
  741
  742
  743          UNIVERSAL_ENTRY SCH$REMOVE_CPU_CAP,<^M<R2,R3,R4,R5>>
  744
  745          MOVL     #1,R5                       ; Indicate REMOVE
  746
  747 1$:      LOCK     MUTEX=SMP$GL_CPU_MUTEX,- ; Lock active set
  748                   PRESERVE=NO,-
  749                   SHARE=YES
  750
  751          INCL     W^SCH$GL_CAPABILITY_SEQUENCE ; Something will change
  752          MOVL     CPU(AP),R4                  ; Get CPU number
  753          BGEQ     5$
  754          BRW      100$                        ; All of them
  755 5$:      BBC      R4,G^SMP$GL_ACTIVE_CPUS,15$  ; Make sure its in active set
```

```
  756          MOVL    G^SMP$GL_CPU_DATA[R4],R3 ; Get CPU data address
  757          MOVL    OLD_MASK(AP),R1          ; Does caller want old mask
  758          BEQL    10$                      ; No
  759          MOVL    CPU$L_CAPABILITY(R3),(R1)
  760 10$:     TSTL    R5               ; ADD or REMOVE
  761          BNEQ    20$                      ; REMOVE
  762          BISL    MASK(AP),CPU$L_CAPABILITY(R3) ; ADD it
  763          BISL    MASK(AP),W^SCH$AL_CPU_CAP[R4] ; ADD it
  764          MOVL    #RSN$_CPUCAP,R0          ; Tell everyone about new capability
  765          JSB     G^SCH$RAVAIL
  766          BRB     30$
  767
  768 15$:     BRW     200$
  769
  770 20$:     BICL    MASK(AP),CPU$L_CAPABILITY(R3) ; REMOVE it
  771          BICL    MASK(AP),W^SCH$AL_CPU_CAP[R4] ; REMOVE it
  772          BSBW    300$                     ; See if RESCHED needed
  773
  774 30$:     MOVL    #SS$_NORMAL,R0
  775 40$:     UNLOCK  MUTEX=SMP$GL_CPU_MUTEX,- ; Unlock active set
  776                  PRESERVE=YES,-
  777                  SHARE=YES
  778          RET
  779
  780 50$:     BRB     30$
  781
  782 ;
  783 ;        Request was made to affect all CPUs in active set  plus default
  784 ;
  785 100$:    MOVL    OLD_MASK(AP),R1          ; Does caller want old mask
  786          BEQL    110$                     ; No
  787          MOVL    G^SCH$GL_DEFAULT_CPU_CAP,(R1)
  788 110$:    TSTL    R5                       ; ADD or REMOVE
  789          BNEQ    120$                     ; REMOVE
  790          BISL    MASK(AP),G^SCH$GL_DEFAULT_CPU_CAP  ; ADD it
  791          MOVL    #RSN$_CPUCAP,R0          ; Tell everyone about new capability
  792          JSB     G^SCH$RAVAIL
  793          BRB     130$
  794
  795 120$:    BICL    MASK(AP),G^SCH$GL_DEFAULT_CPU_CAP  ; REMOVE it
  796
  797 130$:    CLRL    R1                       ; Starting bit for search
  798 140$:    SUBL3   R1,#32,R3                ; Field length for search
  799          FFS     R1,R3,G^SMP$GL_ACTIVE_CPUS,R1 ; Next active CPU
  800          BEQL    50$                      ; Done
  801          MOVL    G^SMP$GL_CPU_DATA[R1],R3 ; Get CPU data address
  802          TSTL    R5                       ; ADD or REMOVE
  803          BNEQ    150$                     ; REMOVE
  804          BISL    MASK(AP),CPU$L_CAPABILITY(R3) ; ADD it
  805          BISL    MASK(AP),W^SCH$AL_CPU_CAP[R1] ; ADD it
  806          INCL    R1                       ; Next
  807          BRB     140$                     ; Continue
  808
  809 150$:    BICL    MASK(AP),CPU$L_CAPABILITY(R3) ; REMOVE it
  810          BICL    MASK(AP),W^SCH$AL_CPU_CAP[R1] ; REMOVE it
  811          BSBB    300$                     ; See if RESCHED needed
  812          INCL    R1                       ; Next
```

```
  813          BRB     140$                    ; Continue
  814
  815 200$:    MOVZWL  #SS$_BADPARAM,R0                ; CPU not in active set
  816          BRW     40$
  817
  818 ;
  819 ;        See if we need to do a RESCHED after removing a capability
  820 ;
  821 300$:    MOVL    CPU$L_CURPCB(R3),R4     ; Current process of affected CPU
  822          BICL3   CPU$L_CAPABILITY(R3),-  ; Can process still run
  823                  PCB$L_CAPABILITY(R4),R0
  824          BEQL    350$                    ; Yes
  825          BICL    #<CPB$M_EXPLICIT_AFFINITY-
  826                  !CPB$M_IMPLICIT_AFFINITY>,R0
  827          BEQL    350$                    ; Still ok if only CPU affinity
  828          FIND_CPU_DATA R4                ; What CPU am I running on?
  829          CMPL    R4,R3                   ; Is it the one I'm affecting
  830          BNEQ    310$                    ; No
  831          SOFTINT #IPL$_RESCHED           ; request a cheap RESCHED
  832          BRB     350$
  833
  834 310$:    IPINT_CPU RESCHED,CPU$L_PHY_CPUID(R3) ; Expensive RESCHED
  835 350$:    RSB
  836
  837          .DSABL LSB
  838
```

```
840          .SBTTL  SCH$CALCULATE_AFFINITY - Calculate current affinity for process
841 ;++
842 ; FUNCTIONAL DESCRIPTION:
843 ;       SCH$CALCULATE_AFFINITY calculates a current affinity mask for the
844 ;       process based on its required capabilities.  The capability sequence
845 ;       mask in the PCB is also updated  so a stale mask can be detected.
846 ;
847 ;       Implicit affinity is ignored in this calculation.
848 ;
849 ; CALLING SEQUENCE:
850 ;       JSB     SCH$CALCULATE_AFFINITY
851 ;
852 ; INPUT PARAMETERS:
853 ;       R4 -> PCB OF AFFECTED PROCESS
854 ;
855 ;
856 ; ENVIRONMENT:
857 ;       The SCHED lock must  be held when the routine is called.
858 ;
859 ;--
860
861 SCH$CALCULATE_AFFINITY::
862
863          MOVQ    R0,-(SP)
864          MOVQ    R2,-(SP)
865          CLRL    PCB$L_CURRENT_AFFINITY(R4)  ; Initialize the CPU affinity
866          MOVL    W^SCH$GL_CAPABILITY_SEQUENCE,-; Save the current capability
867                  PCB$L_CAPABILITY_SEQ(R4)    ;  sequence number
868          MOVL    PCB$L_CAPABILITY(R4),R0
869          ASSUME  CPB$V_EXPLICIT_AFFINITY EQ 31
870          BLSS    200$                        ; Explicit affinity is set
871          BICL    #CPB$M_IMPLICIT_AFFINITY,R0 ; Ignore implicit affinity
872 ;
873 ;       The following code does not lock the  active set.  This is permissable
874 ;       because the capability sequence number can only be updated under the
875 ;       scheduler lock.  The same is true for the declaration of the
876 ;       capability resource.  Thus, even if the active set changes, our
877 ;       incorrect decisions will rapidly be reevaluated.
878 ;
879          MOVL    G^SMP$GL_ACTIVE_CPUS,R1 ; Get copy of active set
880          CLRL    R2                      ; Start at bit 0
881 10$:     SUBL3   R2,#32,R3               ; Bits to check
882          FFS     R2,R3,R1,R2             ; Find an active CPU
883          BEQL    50$                     ; Done
884          BICL3   W^SCH$AL_CPU_CAP[R2],-  ; Can we run here
885                  R0,R3
886          BNEQ    20$                     ; No
887          BBSS    R2,PCB$L_CURRENT_AFFINITY(R4),20$ ; Set the CPU bit
888 20$:     INCL    R2                      ; Next one
889          BRB     10$                     ; Continue
890
891 50$:     MOVQ    (SP)+,R2
892          MOVQ    (SP)+,R0
893          RSB
894
895 200$:    BBCS    PCB$L_AFFINITY(R4),-    ; It can only run on one CPU
896                  PCB$L_CURRENT_AFFINITY(R4),50$
```

```
  897          BRW     QEMPTY                  ; Can't possibly get here!
```

```
SCHED RESCHEDULING INTERRUPT HANDLER 10-MAY-1989 16:39:31 VAX MACRO V5.0-8 Page 23
X-20U6 SCH$ACQUIRE_AFFINITY - Acquire implicit 12-APR-1989 08:40:38 _$254$DUA55:[SYS.SRC]SCH
```

```
899          .SBTTL  SCH$ACQUIRE_AFFINITY - Acquire implicit affinity
900 ;++
901 ; FUNCTIONAL DESCRIPTION:
902 ;       SCH$ACQUIRE_AFFINITY is called to acquire implicit affinity for a
903 ;       specific CPU.
904 ;
905 ; CALLING SEQUENCE:
906 ;       CALLS  #3,SCH$ACQUIRE_AFFINITY
907 ;
908 ; INPUT PARAMETERS:
909 ;       arg1 =>  PCB of the affected process
910 ;
911 ;       arg2 =   Routine to be called on breaking affinity
912 ;
913 ;       arg3 =   CPU number
914 ;
915 ;
916 ; ENVIRONMENT:
917 ;       The SCHED lock must be held.
918 ;
919 ;
920 ; CALLBACK (CALLS interface)
921 ;       arg1 =>  PCB of the affected process
922 ;
923 ;       arg2 =   CPU number
924 ;
925 ;
926 ;
927 ;--
928
929          PCB     = 4
930          CALLBACK= 8
931          CPU     = 12
932
933          UNIVERSAL_ENTRY SCH$ACQUIRE_AFFINITY,<^M<R4>>
934
935          MOVL    PCB(AP),R4                      ; Get PCB address
936          TSTL    PCB$L_AFFINITY_CALLBACK(R4)     ; Is there already one?
937          BNEQ    500$                            ; Yes
938          ASSUME CPB$V_EXPLICIT_AFFINITY  EQ  31
939          TSTL    PCB$L_CAPABILITY(R4)            ; Is there explicit affinity?
940          BGEQ    20$                             ; No
941          CMPL    CPU(AP),PCB$L_AFFINITY(R4)      ; Do they match
942          BNEQ    500$                            ; No - error
943 20$:     MOVL    CALLBACK(AP),-                  ; Record callback
944                  PCB$L_AFFINITY_CALLBACK(R4)
945          BISL    #CPB$M_IMPLICIT_AFFINITY,-      ; Request implicit affinity
946                  PCB$L_CAPABILITY(R4)
947          MOVL    CPU(AP),PCB$L_AFFINITY(R4)      ; Record the CPU
948          MOVB    G^SCH$GL_AFFINITY_SKIP,-        ; Set skip count
949                  PCB$B_AFFINITY_SKIP(R4)
950          CMPW    PCB$W_STATE(R4),#SCH$C_CUR      ; Is it executing
951          BNEQ    100$                            ; No - done
952          MOVL    PCB$L_CPU_ID(R4),R0             ; Get current CPU
953          CMPL    R0,CPU(AP)                      ; Is it the right place
954          BEQL    100$                            ; Yes
955          FIND_CPU_DATA R1                        ; Get CPU data
```

```
  956          CMPL    R0,CPU$L_PHY_CPUID(R1)          ; Is it us
  957          BNEQ    50$                             ; No
  958          SOFTINT #IPL$_RESCHED                   ; Yes  - resched
  959          BRB     100$
  960 50$:     IPINT_CPU RESCHED,R0                    ; Resched the current CPU
  961 100$:    MOVL    #SS$_NORMAL,R0
  962          RET
  963
  964 500$:    MOVL    #SS$_BADPARAM,R0                ; Error
  965          RET
```

```
  967          .SBTTL  SCH$REMOVE_AFFINITY - Remove implicit affinity
  968 ;++
  969 ; FUNCTIONAL DESCRIPTION:
  970 ;        SCH$REMOVE_AFFINITY is called to remove implicit affinity for a
  971 ;        specific CPU.
  972 ;
  973 ; CALLING SEQUENCE:
  974 ;        CALLS  #3,SCH$REMOVE_AFFINITY
  975 ;
  976 ; INPUT PARAMETERS:
  977 ;        arg1 =>  PCB of the affected process
  978 ;
  979 ;        arg2 =   Callback routine specified in SCH$ACQUIRE_AFFINITY
  980 ;
  981 ;        arg3 =   CPU number
  982 ;
  983 ;
  984 ; ENVIRONMENT:
  985 ;        The SCHED lock must be held.
  986 ;
  987 ;
  988 ;--
  989
  990          PCB     = 4
  991          CALLBACK= 8
  992          CPU     = 12
  993
  994          UNIVERSAL_ENTRY SCH$REMOVE_AFFINITY,<^M<R4>>
  995
  996          MOVL    PCB(AP),R4                      ; Get PCB address
  997          CMPL    CALLBACK(AP),-                  ; Do routine addresses match
  998                  PCB$L_AFFINITY_CALLBACK(R4)
  999          BNEQ    500$                            ; No
 1000          CLRL    PCB$L_AFFINITY_CALLBACK(R4);    ; Cancel the request
 1001          BBC     #CPB$V_IMPLICIT_AFFINITY,-      ; Should we keep implicit aff?
 1002                  G^SCH$GL_DEFAULT_PROCESS_CAP,100$ ; Done if so
 1003          BICL    #CPB$M_IMPLICIT_AFFINITY,-      ; Clear it
 1004                  PCB$L_CAPABILITY(R4)
 1005 100$:   MOVL    #SS$_NORMAL,R0
 1006          RET
 1007
 1008 500$:   MOVL    #SS$_BADPARAM,R0                ; Error
 1009          RET
 1010
 1011          .END
```

4 SHELL.LIS

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 0
Table of contents

  (3)       450        Null pagefile
  (4)       474        SHELL PROCESS HEADER
  (5)       616        SHELL PROCESS PAGE TABLES
  (6)       755        BODY OF SHELL PROCESS
  (6)      1123        PROCESS I/O SEGMENT
  (6)      1368        INITIALZE SHELL WITH SYSGEN PARAMETERS
  (6)      1644        FILLPHD - SETUP A VALID PHD PTE
  (6)      1722        COMMAND LANGUAGE INTERPRETER DATA AREA
  (6)      1759        COMPATIBILITY MODE EMULATOR CONTEXT PAGE
  (6)      1768        GLOBAL USER-MODE CONTEXT PAGE
  (6)      1800        IMAGE ACTIVATOR CONTEXT PAGE
  (6)      1893        Kernel mode, process OWN data pages
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 1
X-45U3 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (2)

    2 .IF DF,LIBSWITCH
    3         .TITLE  SYS$USRVECTOR - USER ACCESSIBLE P1 CELLS
    4 .IFF
    5         .IF DF PRMSW
    6         .TITLE  PROCESS_DEFINITIONS_MASK
    7         .IFF
    8         .IF     NDF,PROCESS_PAGE_DEFINITIONS
    9         .TITLE  SHELL
   10         .IFF
   11         .TITLE  PROCESS_PAGE_DEFINITIONS
   12         .ENDC
   13         .ENDC
   14 .ENDC
   15         .IDENT  'X-45U3'
   16 ;
   17 ;******************************************************************************
   18 ;*                                                                            *
   19 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984, 1987 BY                             *
   20 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
   21 ;*  ALL RIGHTS RESERVED.                                                      *
   22 ;*                                                                            *
   23 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
   24 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
   25 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
   26 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
   27 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
   28 ;*  TRANSFERRED.                                                              *
   29 ;*                                                                            *
   30 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
   31 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
   32 ;*  CORPORATION.                                                              *
   33 ;*                                                                            *
   34 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
   35 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
   36 ;*                                                                            *
   37 ;*                                                                            *
   38 ;******************************************************************************
   39 ;
   40 ;++
   41 ; FACILITY:     Executive, create process/swapper data base
   42 ;
   43 ; ABSTRACT:     Shell defines the swap image for the initial process
   44 ;               started by the create process system service.
   45 ;
   46 ;               This module produces four object modules:
   47 ;
   48 ;               1)      The shell used initialize the minimal
   49 ;                       process context, which is part of the
   50 ;                       working_set_management image.
   51 ;
   52 ;               2)      An object module containing global
   53 ;                       definitions of p1 vector page cells
   54 ;                       and other miscellanous assembly
   55 ;                       definitions needed by other parts of
   56 ;                       the exec.  This module is linked into
   57 ;                       the base image (sys.exe), and produces no
   58 ;                       no storage.
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 2
X-45U3 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (2)

   59 ;
   60 ;               3)      An object module containing global symbol
   61 ;                       definitions which can be used by any program
   62 ;                       which does not link against sys.stb.
   63 ;
   64 ;               4)      An object module containing version mask
   65 ;                       values, which will be linked into sys to
   66 ;                       produce double valued version symbols for
   67 ;                       all global symbols in this module.
   68 ; SIDE EFFECTS:
   69 ;
   70 ;               When making changes to this module, make sure
   71 ;               the first two assemblies produce the same global symbol
   72 ;               definitions.
   73 ;
   74 ; ENVIRONMENT:
   75 ;
   76 ;               The initialization of the minimal process context for
   77 ;               a newly created process is executed under the context
   78 ;               of the swapper.
   79 ;
   80 ; AUTHOR: RICHARD I. HUSTVEDT, CREATION DATE: 30-NOV-76
   81 ;
   82 ; MODIFIED BY:
   83 ;
   84 ;       X-45U3  LSS0080         Leonard S. Szubowicz    06-Oct-1988
   85 ;               Add latent support cells for RMS/DDTM, namely:
   86 ;               PIO$GL_RM0_RM_ID alias for PIO$GL_RU_HANDLER_ID,
   87 ;               PIO$GQ_RUB_LH alias for PIO$GL_RUB_FLINK/BLINK,
   88 ;               cells PIO$GL_NT0_RM_ID, PIO$GQ_NTRUB_LH, PIO$GQ_RUF_TSB_LH,
   89 ;               PIO$GL_RESERVED0 through 6.
   90 ;
   91 ;       X-45U2  WMC0U2          Wayne Cardoza           21-Sep-1988
   92 ;               Add 2 POSIX cells.
   93 ;
   94 ;       X-45U1  JEJ0524         James E Johnson         27-Jun-1988
   95 ;               Add latent support cells for DDTM.  Notably:
   96 ;               CTL$GQ_XCB_QUE, CTL$GQ_RMCB_QUE, CTL$GL_CUR_XSCB
   97 ;
   98 ;       X-45    JDC0385         Jon Callas              23-FEB-1988
   99 ;               Rearrange a couple pages so they don't get deleted while
  100 ;               ASTs can still reference them.
  101 ;
  102 ;       X-44    WMC0044         Wayne Cardoza           29-Dec-1987
  103 ;               Add version categories.
  104 ;
  105 ;       X-43    SF00043         Stephen Fiorelli        20-Nov-1987
  106 ;               Collect version masks into the proper psects.
  107 ;
  108 ;       X-42    SF00042         Stephen Fiorelli        11-Nov-1987
  109 ;               More changes for version stuff.  Shorten the name
  110 ;               of this module for the version assembly so the
  111 ;               librarian can deal with it.  Don't assemble version
  112 ;               values for mmg$ar_nullpfl and swp$gl_shellbas since
  113 ;               they are vectored values and have already been
  114 ;               defined in system_data_cells.
  115 ;
```

```
116 ;       X-41    SSA0013         Stan Amway              28-Oct-1987
117 ;               Charge page file quota for kernel stack expansion
118 ;               pages by including them in the symbolic constant
119 ;               SWP$C_SHELLPFIL. Not doing this will (generally)
120 ;               cause the delete page logic to erroneously return
121 ;               page file quota that was never deducted.
122 ;
123 ;       X-40    WMC0040         Wayne Cardoza           15-Oct-1987
124 ;               Add more version stuff.
125 ;
126 ;       X-39    WMC0039         Wayne Cardoza           12-Oct-1987
127 ;               New assembly for version masks.
128 ;
129 ;       X-38    HH0300          Hai Huang               09-Oct-1987
130 ;               Add per-process (UW) cells for VVIEF support.
131 ;
132 ;       X-37    JLV0001         Jake VanNoy             31-Aug-1987
133 ;               Add two new byte definitions used for fast $SETAST
134 ;               equivalent.  CTL$GB_SOFT_AST_DISABLE is set to 1
135 ;               from user mode code to disable user-mode AST delivery.
136 ;               CTL$GB_REENABLE_ASTS is read at end of critical
137 ;               section of user mode code (when disable is cleared),
138 ;               If set, the code should call $SETAST(1) because ASTs
139 ;               were actually disabled when an AST was queued to the
140 ;               process.  (Also add 2 page VWS area)
141 ;
142 ;       X-36    SF00036         Stephen Fiorelli        04-Sep-1987
143 ;               Final large working set support.
144 ;
145 ;       X-35    JWT0298         Jim Teague              27-Aug-1987
146 ;               Define a longword for PROCESS START FLAGS.
147 ;
148 ;       X-34    JDC0378         Jon Callas              25-AUG-1987
149 ;               Add in a quadword of flags to the P1 data region.
150 ;
151 ;       X-33    SSA0012         Stan Amway              24-Aug-1987
152 ;               Rework the logic that establishes the working set
153 ;               related quotas to track changes in PROCSTRT and
154 ;               INITUSER.
155 ;
156 ;       X-32    SSA0011         Stan Amway              10-Aug-1987
157 ;               Final swap file allocation changes.
158 ;               Limit WSQUOTA to 64K pages.
159 ;
160 ;       X-31    SF00031         Stephen Fiorelli        7-Aug-1987
161 ;               WSFLUID cell in the process header is extended to
162 ;               a longword.
163 ;
164 ;       X-30    LSS0055         Leonard S. Szubowicz    06-Aug-1987
165 ;               Add PIO$GB_JNL_STALL_CNT cell for use by RMS.
166 ;
167 ;       X-29    SJF             Stu Farnham             27-Jul-1987
168 ;               INVALID becomes INVALIDATE_TB. Make TBIS track SRM
169 ;               rev H.
170 ;
171 ;       X-28    SF00028         Stephen Fiorelli        17-Jul-1987
172 ;               Initial large working set support.
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 4
X-45U3 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (2)

  173 ;
  174 ;       X-27    SSA0010         Stan Amway              23-Jun-1987
  175 ;               Preliminary swap file allocation changes.
  176 ;
  177 ;               WMC0027         Wayne Cardoza           01-Jul-1987
  178 ;               Decrease PHD$L_FREPTECNT by 7 to ensure padded page tables.
  179 ;
  180 ;       X-26    SSA0009         Stan Amway               5-May-1987
  181 ;               Increase SWP$C_SHLP1PT from 2 to 3 pages. LSS0026
  182 ;               caused the number of P1 pages to exceed 256.
  183 ;
  184 ;       X-25    SSA0008         Stan Amway               9-Apr-1987
  185 ;               Change NULL pagefile control block to track changes
  186 ;               in $PFLDEF. Do it in such a way that such manual
  187 ;               tracking will not be required in the future.
  188 ;
  189 ;       X-24    SSA0007         Stan Amway               1-Apr-1987
  190 ;               Account for process page table pages in PPGFLVA.
  191 ;
  192 ;       X-23    RNG5023         Rod Gamache             16-Mar-1987
  193 ;               Initialize CTL$GL_PCB in SHELL, rather than PROCSTRT.
  194 ;
  195 ;       X-22    SSA0006         Stan Amway              16-Mar-1987
  196 ;               Adjust PPGFLVA to account for mapped, SHELL pages.
  197 ;
  198 ;       X-21    SSA0005         Stan Amway               5-Mar-1987
  199 ;               Change computation of initial, prereserved pagefile
  200 ;               pages requirement to correctly reflect process
  201 ;               header pages.
  202 ;
  203 ;       X-20    SF04003         Stephen Fiorelli        05-Feb-1987
  204 ;               Move the null pagefile from the base image to this
  205 ;               module.
  206 ;
  207 ;       X-19    LSS0026         Leonard S. Szubowicz    11-Feb-1987
  208 ;               Increase the size of the RMS directory cache PIO$A_DIRCACHE
  209 ;               from 2 to 4 pages and delete the RMS tracepoint page PIO$A_TRACE
  210 ;               Add global symbols PIO$A_RMS_PIOBASE and PIO$A_RMS_PIOEND to
  211 ;               delimit fixed RMS PIO region.
  212 ;
  213 ;       X-18    SSA0004         Stan Amway              28-Nov-1987
  214 ;               Slight change in call sequence to MMG$ASNPRCPGFLP.
  215 ;
  216 ;       X-17    LMP0439         L. Mark Pilant,         28-JAN-1987  10:01
  217 ;               Add storage for protected subsystem identifiers.
  218 ;
  219 ;       X-16    SF04002         Stephen Fiorelli        26-Jan-1987
  220 ;               Minor changes necessary to make the shell pageable.
  221 ;               SWP$GL_SHELLBAS is vectored, and put the shell into
  222 ;               a pageable read only psect.  Initialization of the
  223 ;               null pagefile structure occurs in INIT.
  224 ;
  225 ;       X-15    DDP0032         Derrell D. Piper        26-Jan-1987
  226 ;               Add CTL$GQ_JPICTX, CTL$GL_JPICTXCNT, and CTL$GL_UAICTX
  227 ;               for use by $GETJPI, $GETUAI, and $SETUAI.
  228 ;
  229 ;       X-14    DAS             David A. Solomon        25-Jan-1987
```

```
230 ;               Add CTL$GL_RMS_RUJLCK_FLINK/BLINK (RMS RU journal lock blocks).
231 ;
232 ;       X-13    SSA0003         Stan Amway              12-Nov-1986
233 ;               Support for multiple pagefiles per process.
234 ;               Add 2 demand zero pages for kernel, process OWN data.
235 ;               Add cell to P1 pointer page to contain address of
236 ;               kernel data extension area (future use).
237 ;
238 ;       X-12    DAS             David A. Solomon        06-Dec-1986
239 ;               Remove PIO$GW_LNKCSHCNT. This was being used as an RMS RUNDOWN
240 ;               in progress FLAG FOR THE NETWORK CODE. THE NETWORK CODE NOW
241 ;               checks IMP$V_IORUNDOWN.
242 ;
243 ;       X-11    DAS             David A. Solomon        13-Nov-1986
244 ;               Change the RMS RUB structure listhead (PIO$GL_RULOCK) to be
245 ;               a queue header (PIO$GL_RUB_FLINK/BLINK).
246 ;
247 ;       X-10    SSA0002         Stan Amway               7-Nov-1986
248 ;               Add ASSUME to catch PHD too large.
249 ;
250 ;       X-9     SSA0001         Stan Amway              16-Oct-1986
251 ;        -8     Support for exec mode rundown routines.
252 ;
253 ;       X-7     PMV0009         Peter M. Vatne          22-Sep-1986
254 ;               Rename PIO$GL_RU_HAN_ID to PIO$GL_RU_HANDLER_ID.
255 ;
256 ;       X-6     JEJ0360         James E Johnson          7-Sep-1986
257 ;               Fix typo in previous edit.
258 ;
259 ;       X-5     JEJ0359         James E Johnson          5-Sep-1986
260 ;               Add PIO and CTL cells for RMS Journaling and logical link
261 ;               caching.
262 ;
263 ;               SF04001         Stephen Fiorelli        01-Aug-1986
264 ;               Changes required in the movement of the shell into
265 ;               the image containing the swapper (working_set_management).
266 ;
267 ;               1)      References to MMDAT cells (local copy of read-only
268 ;                       memory management cells) changed to reference the
269 ;                       base image version of these cells.
270 ;
271 ;               2)      Previous conditional assemblies torn out and replaced
272 ;                       by new ones that produce the modules described in
273 ;                       the abstract.
274 ;
275 ;               3)      Elimination of the cell SWP$GL_SHELIO.  Cell is
276 ;                       no longer needed, an assembly time constant will
277 ;                       be used in its place.
278 ;
279 ;       Include files
280 ;
281
282         $DYNDEF                 ;DYNAMIC DATA STRUCTURE TYPE DEFINITIONS
283         $FWADEF                 ;RMS FILE WORK AREA CONSTANTS
284         $IAFDEF                 ;IMAGE ACTIVATOR FIXUP VECTOR OFFSETS
285         $IMPDEF                 ;RMS INTERNAL STRUCTURE DEFINITION
286         $IPLDEF                 ;DEFINE INTERRUPT LEVELS
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 6
X-45U3 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (2)

  287         $JIBDEF                 ;JOB INFORMATION BLOCK DEFINITIONS
  288         $NSAIDTDEF              ;SECURITY AUDITING IMPURE DATA TABLE OFFSETS
  289         $PCBDEF                 ;PROCESS CONTROL BLOCK DEFINITIONS
  290         $PFLDEF                 ;PAGE FILE DEFINITIONS
  291         $PFNDEF                 ;PFN DATA BASE DEFINITIONS
  292         $PHDDEF                 ;PROCESS HEADER DEFINITIONS
  293         $PQBDEF                 ;PROCESS QUOTA BLOCK DEFINITIONS
  294         $PRTDEF                 ;DEFINE PROTECTION CODES
  295         $PSLDEF                 ;DEFINE PSL FIELDS
  296         $PTEDEF                 ;PAGE TABLE ENTRY DEFINTIONS
  297         $SECDEF                 ;SECTION DEFINITIONS (GSTE/PSTE)
  298         $SGNDEF                 ;SYSGEN VALUE DEFINITIONS
  299         $VADEF                  ;DEFINE VIRTUAL ADDRESS FIELDS
  300         $WSLDEF                 ;WORKING SET LIST DEFINITIONS
  301 .IF DF PRMSW
  302         $SYSVERSIONDEF
  303 .ENDC
  304
  305         .MACRO  DEFINE_CONSTANT   SYM,CONSTANT,VERSION_MASK
  306         .IF NDF PRMSW
  307                 SYM == CONSTANT
  308         .IFF
  309                 SYM == 0
  310                 .IRP    ...SUBVERSION...,<VERSION_MASK>
  311                 SYM == SYM!<1@SYS$K_'...SUBVERSION...>
  312                 .ENDR
  313         .ENDC
  314         .ENDM
  315
  316 ;
  317 ; EQUATES:
  318 ;
  319
  320         PFILPGCNT=0                             ; INITIALIZE COUNT OF PAGE FILE PAGE
  321         PFILPGCNT_NDZRO=0                       ; Init count of page file pages
  322                                                 ; (not demand zero)
  323         KSTACK=3                                ; THREE PAGES OF KERNEL STACK
  324         KSTACK_EX=4                             ; KERNEL STACK EXPANSION PAGES
  325         ESTACK=16                               ; SIXTEEN PAGES OF EXEC STACK
  326         SSTACK=32                               ; THIRTY TWO PAGES OF SUPER STACK
  327         IMGACTBUF=8                             ; EIGHT PAGES FOR IMAGE ACTIVATE BUF
  328                                                 ; FOUR FOR TOP LEVEL CALL, FOUR FOR
  329                                                 ; ONE RECURSIVE CALL.
  330         KRP_COUNT=4                             ; SIZE IN PAGES OF P1 LOOKASIDE LIST
  331
  332         DBGPTCNT=1                              ; NUMBER OF DEBUG AREA PAGE TABLES
  333
  334 .IF NDF LIBSWITCH                               ; DON'T ASSEMBLE FOR USRVECTOR MODUL
  335         DEFINE_CONSTANT SWP$C_DBGPTCNT,DBGPTCNT,STABLE
  336         DEFINE_CONSTANT SWP$C_KSTACK,KSTACK,MISC
  337         DEFINE_CONSTANT SWP$C_KSTACK_EX,KSTACK_EX,MISC
  338         DEFINE_CONSTANT CTL$C_KRP_COUNT,KRP_COUNT,MISC
  339         DEFINE_CONSTANT CTL$C_KRP_SIZE,512,MISC ; EACH KRP PACKET IS 512 BYTES LONG
  340 .ENDC
  341
  342         COMMON=4                                ; FOUR PAGES FOR USER COMMON
  343                                                 ; FOUR PAGES FOR DEC COMMON
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 7
X-45U3 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (2)

  344
  345 ;
  346 ; MACROS:
  347 ;
  348
  349          .MACRO  WSL     SYM,TYPE        ; MAKE VALID WS LIST ENTRY
  350          .LONG   SYM!WSL$M_VALID!WSL$C_'TYPE     ; SET VALID BIT
  351          .ENDM   WSL
  352
  353          .MACRO  CTL     SYM,VERSION_MASK
  354 .IF NDF,LIBSWITCH                        ; NO GLOBALS FOR USRVECTOR MODULE
  355 .IF NDF PRMSW
  356          CTL$'SYM==.-CTLVECPAG+VECTORS
  357 .IFF
  358          CTL$'SYM== 1@SYS$K_SHELL
  359          .IRP    ...SUBVERSION...,<VERSION_MASK>
  360          CTL$'SYM == CTL$'SYM!<1@SYS$K_'...SUBVERSION...>
  361          .ENDR
  362 .ENDC
  363 .IFF
  364          CTL$'SYM=.-CTLVECPAG+VECTORS
  365 .ENDC
  366          .ENDM   CTL
  367
  368          .MACRO  PHD     SYM
  369          .=PHD...+PHD$'SYM
  370          .ENDM   PHD
  371
  372          .MACRO  PFL     SYM,VALUE
  373          .show meb
  374          .=MMG$AR_NULLPFL+PFL$'SYM
  375          .IIF    IDN,<%EXTRACT(0,1,SYM)>,<B>, .BYTE      VALUE
  376          .IIF    IDN,<%EXTRACT(0,1,SYM)>,<W>, .WORD      VALUE
  377          .IIF    IDN,<%EXTRACT(0,1,SYM)>,<L>, .LONG      VALUE
  378          .noshow meb
  379          .ENDM   PFL
  380
  381
  382          .MACRO  PIO     SYM,VERSION_MASK  ; DEFINE PROCESS I/O SEGMENT SYMBOL
  383 .IF NDF PRMSW
  384          PIO$'SYM==PIOBASE+<.-PIO>               ;
  385 .IFF
  386          PIO$'SYM== 1@SYS$K_SHELL
  387          .IRP    ...SUBVERSION...,<VERSION_MASK>
  388          PIO$'SYM == PIO$'SYM!<1@SYS$K_'...SUBVERSION...>
  389          .ENDR
  390 .ENDC
  391          .ENDM   PIO
  392
  393          .MACRO  PIOL    SYM             ; DEFINE PROCESS I/O LOCAL SYMBOL
  394 PIOL'SYM=PIOBASE+<.-PIO>                 ;
  395          .ENDM   PIOL                    ;
  396
  397          .MACRO  PIO2    SYM             ; DEFINE PROCESS I/O LOCAL SYMBOL
  398 PIO2'SYM=PIOTBL+<.-TBL>
  399          .ENDM   PIO2
  400
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 8
X-45U3 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (2)

  401
  402          .MACRO  DEFINE_SYMBOL   SYM,VERSION_MASK
  403          .IF NDF PRMSW
  404                  SYM == .
  405          .IFF
  406                  SYM == 1@SYS$K_SHELL
  407                  .IRP    ...SUBVERSION...,<VERSION_MASK>
  408                  SYM == SYM!<1@SYS$K_'...SUBVERSION...>
  409                  .ENDR
  410          .ENDC
  411          .ENDM
  412
  413 ;
  414 ; P1PTE - GENERATE PTES FOR CONTROL REGION PAGE TABLE
  415 ;
  416
  417          .MACRO  P1PTE   NUM,ACCESS=NA,OWNER=K,STATE,SYM
  418          .IF     NB,SYM
  419          SYM=VA...
  420          .ENDC
  421
  422          .REPT   NUM
  423          .=.-4
  424          .IF     IDN,<STATE>,<VALID>
  425          TMP...=<1@31>
  426          .IFF
  427          TMP...=0
  428          .ENDC
  429
  430          .IF     IDN,<STATE>,<DZRO>
  431          PFILPGCNT=PFILPGCNT+1
  432          .ENDC
  433
  434          .IF     IDN,<STATE>,<PFIL>
  435          TMP...=PTE$M_TYP1
  436          PFILPGCNT=PFILPGCNT+1
  437          PFILPGCNT_NDZRO=PFILPGCNT_NDZRO+1
  438          .ENDC
  439          .LONG   PTE$C_'ACCESS!PTE$C_'OWNER'OWN!TMP...
  440
  441          .=.-4
  442          VA...=VA...-^X200
  443          .ENDR
  444          .ENDM   P1PTE
  445
  446 .IF NDF,PRMSW                        ; Don't assemble for version masks
  447 .IF NDF,LIBSWITCH                    ; FOR USRVECTOR MODULE
  448          .IF     NDF,PROCESS_PAGE_DEFINITIONS
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 9
X-45U3 Null pagefile 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (3)

  450          .SBTTL  Null pagefile
  451 ;+
  452 ;        The null pagefile structure serves two purposes.
  453 ;        First, it serves as a placeholder in the pagefile
  454 ;        vector.  Second it is used to locate the shell
  455 ;        within the working_set_management image, when it
  456 ;        is necessary to swap the shell in.
  457 ;+
  458          DECLARE_PSECT   EXEC$NONPAGED_DATA,ALIGNMENT=LONG
  459
  460          UNIVERSAL_SYMBOL MMG$AR_NULLPFL
  461          .BYTE   0[PFL$C_LENGTH]          ; Reserve pre-zeroed space for structure


  462
  463 ;
  464 ; Initialize non-zero fields of structure
  465 ;
  466          PFL     W_SIZE,PFL$C_LENGTH      ; Structure size
               .=MMG$AR_NULLPFL+PFL$W_SIZE
               .IIF    IDN,<W>,<W>, .WORD       PFL$C_LENGTH
  467          PFL     B_TYPE,DYN$C_PFL         ; Structure type
               .IIF    IDN,<B>,<B>, .BYTE       DYN$C_PFL
  468          PFL     L_MAXVBN,PTE$M_PGFLVBN   ; Page file VBN mask
               .=MMG$AR_NULLPFL+PFL$L_MAXVBN
               .IIF    IDN,<L>,<L>, .LONG       PTE$M_PGFLVBN
  469
  470          .ENDC
  471 .ENDC
  472 .ENDC
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 10
X-45U3 SHELL PROCESS HEADER 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (4)

  474         .SBTTL  SHELL PROCESS HEADER
  475 .IF DF,LIBSWITCH                        ; FOR USRVECTOR MODULE
  476         .PSECT  $ABS$,ABS               ; DO NOT GENERATE ANY STORAGE
  477         .=0
  478 .IFF                                    ; EXCLUDE USRVECTOR MODULE
  479 .IF NDF,PRMSW                           ; for version masks
  480         .IF     DF,PROCESS_PAGE_DEFINITIONS
  481         .PSECT  $ABS$,ABS               ; DO NOT GENERATE ANY STORAGE
  482         .=0
  483         .IFF                            ; ELSE,
  484         DECLARE_PSECT   EXEC$PAGED_CODE,ALIGNMENT=PAGE
  485         UNIVERSAL_SYMBOL SWP$GL_SHELLBAS
  486         .ENDC
  487 .IFF
  488         .PSECT  $ABS$,ABS               ; If version mask assembly
  489 .ENDC
  490 ;
  491 ;       WARNING: SWP$C_SHLP1PT MUST BE MANUALLY COMPUTED SINCE IT IS USED
  492 ;                TO ALLOCATE THE SPACE FOR P1 PAGE TABLE ENTRIES.  IF UPDATES
  493 ;                ARE MADE THAT INTRODUCE MORE P1 PAGE TABLE ENTRIES, THEN
  494 ;                VERIFY THAT THEY STILL FIT IN THE NUMBER OF PAGES PROVIDED
  495 ;
  496                                 ; COUNT OF BASIC P1 PAGE TABLES FOR SHELL
  497 DEFINE_CONSTANT SWP$C_SHLP1PT,3,MEMORY_MANAGEMENT
  498 SHELL:                                  ; BASE OF SHELL SWAP IMAGE
  499 ;
  500 ;       PROCESS HEADER FOR SHELL PROCESS
  501 ;
  502
  503 PHD...=.                               ; BASE OF PROCESS HEADER
  504         .BLKB   PHD$C_LENGTH            ; RESERVE SPACE FOR HEADER
  505
  506 ;
  507 ;       WORKING SET LIST
  508 ;
  509 ; PAGES DESCRIBED IN THE WORKING SET LIST MUST OCCUR IN THE SAME ORDER
  510 ; WITHIN THE PROCESS BODY.
  511 ;
  512 .IF     NDF     PRMSW
  513 .IF     NDF     PROCESS_PAGE_DEFINITIONS
  514         .ALIGN  LONG                    ; AT LONGWORD RESOLUTION
  515 .IFF
  516         .=<.+3>&^XFFFC
  517 .ENDC
  518 .ENDC
  519 DEFINE_CONSTANT SWP$C_KSTACK_EX_WSL,<<.-PHD...>@-2>,MISC
  520         .BLKL   KSTACK_EX               ; EXTRA SLOTS FOR KERNEL STACK EXPANSION
  521 WSL...=.                               ; BASE OF WORKING SET LIST
  522 DEFINE_CONSTANT SWP$C_KSTACK_WSL,<<.-PHD...>@-2>,MISC
  523         WSL     <<KSPINI-<1*512>>!WSL$M_WSLOCK>,PROCESS; KERNEL STACK PAGE 1
  524         WSL     <<KSPINI-<2*512>>!WSL$M_WSLOCK>,PROCESS; KERNEL STACK PAGE 2
  525         WSL     <<KSPINI-<3*512>>!WSL$M_WSLOCK>,PROCESS; KERNEL STACK PAGE 3
  526 NOT_KSTACK_WSL=.                        ; REST OF WSL IS NOT KERNEL STACK
  527         WSL     <VECTORS!WSL$M_WSLOCK>,PROCESS  ; VECTOR PAGE
  528 DYNWSL=.                               ; START OF DYNAMIC WORKING SET
  529         WSL     <PIOBASE!WSL$M_MODIFY>,PROCESS  ; PROCESS I/O SEGMENT BASE PAGE 1
  530
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 11
X-4503 SHELL PROCESS HEADER 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (4)

  531 FWSL...=.                                 ; FREE WORKING SET LIST
  532 NDYN=<FWSL...-DYNWSL>@-2                  ; NUMBER OF DYNAMIC ENTRIES IN SHELL
  533 SHUFFLE_COUNT=<FWSL...-NOT_KSTACK_WSL>@-2 ; AMOUNT OF SHUFFLING BY SHELINIT
  534 DEFINE_CONSTANT SWP$C_NDYN,NDYN,MISC      ; GLOBAL NAME FOR NUMBER OF DYNAMIC PAGES
  535
  536 .IF      NDF     PRMSW
  537 .IF      NDF     PROCESS_PAGE_DEFINITIONS
  538         .ALIGN  PAGE                      ; GET TO END OF PAGE
  539 .IFF
  540         .=<.+511>&^XFE00
  541 .ENDC
  542 .ENDC
  543
  544 ;*****************************************************************************
  545 ; NB: Until SWP$SHELINIT is generalized, this ASSUME statement must remain. *
  546 ;*****************************************************************************
  547         .IF NDF PRMSW
  548         ASSUME <. - PHD...> LE 512
  549         .ENDC
  550
  551
  552 SAV...=.                                   ;
  553
  554 TMP...=<DYNWSL-PHD...>@-2                 ; BASE OF FLUID WORKING SET LIST
  555
  556         PHD     L_WSLOCK                  ; POINTER TO START OF LOCKED PAGES
  557         .LONG   TMP...                    ;
  558
  559         PHD     L_WSDYN                   ; POINTER TO START OF DYNAMIC PAGES
  560         .LONG   TMP...                    ;
  561
  562         PHD     L_WSLIST                  ;
  563         .LONG   <WSL...-PHD...>@-2        ; START OF WORKING SET LIST
  564         PHD     L_WSNEXT                  ; NEXT WORKING SET ENTRY
  565         .LONG   <TMP...+NDYN-1>           ;
  566
  567         PHD     L_FREP1VA                 ; VA OF FIRST FREE PAGE IN P1 SPACE
  568         .LONG   VA...-512                 ;
  569
  570         PHD     Q_PRIVMSK                 ; ENABLE ALL PRIVILEGES
  571         .LONG   -1,-1                     ;
  572
  573         PHD     W_QUANT                   ; QUANTUM OF ONE SECOND
  574         .WORD   -100                      ;
  575
  576         PHD     L_KSP                     ; STACK POINTERS
  577         .LONG   KSPINI                    ; KERNEL STACK POINTER
  578         .LONG   ESPINI                    ; EXECUTIVE STACK POINTER
  579         .LONG   SSPINI                    ; SUPER STACK POINTER
  580                                           ; NO INITIAL USER STACK
  581
  582         PHD     L_PC                      ; INITIAL PROGRAM COUNTER
  583         .LONG   EXE$PROCSTRT              ; START PROCESS
  584
  585         PHD     L_PSL                     ; PROGRAM STATUS LONGWORD
  586         .LONG   IPL$_ASTDEL@PSL$V_IPL     ; MUST RUN AT ASTDEL
  587         PHD     L_P0BR                    ; P0 BASE REGISTER
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 12
X-45U3 SHELL PROCESS HEADER 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (4)

  588          .LONG   0                       ; RELATIVE OFFSET
  589
  590          PHD     L_P0LRASTL              ; NO AST PENDING, P0LR = 0
  591          .LONG   <4@PHD$V_ASTLVL>        ;
  592
  593          PHD     L_P1BR                  ; SVA OF P1 PAGE TABLE BASE
  594          .LONG   -<1@23>                 ;
  595
  596          PHD     L_P1LR                  ; P1 LENGTH REGISTER
  597          .LONG   <1@21>-P1PTLEN          ;
  598
  599          PHD     L_FREPTECNT             ; FREE PTE'S BETWEEN P0 AND P1 PT
  600          .LONG   -<1+<<1@31-VA...>@-9>>  ; (NUMBER OF AVAILABLE LONGWORDS
  601
  602          PHD     W_PRCLM                 ; SUB PROCESS LIMIT
  603          .WORD   4                       ; ALLOW FOUR
  604
  605          PHD     L_WSFLUID               ; GUARANTEED FLUID PAGE COUNT
  606          .LONG   10                      ; GUESS AT TEN FOR NOW (WAG)
  607
  608          PHD     B_DFPFC                 ; DEFAULT PAGE FAULT CLUSTER
  609          .BYTE   16                      ;
  610
  611          PHD     B_CPUMODE               ; ACCESS MODE FIELD FOR CPU TIME LIMIT
  612          .BYTE   PSL$C_USER              ;  EXPIRATION AST STARTS WITH USER MODE
  613
  614          .=SAV...                        ; RESTORE LOCATION COUNTER
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 13
X-45U3 SHELL PROCESS PAGE TABLES 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (5)

  616          .SBTTL  SHELL PROCESS PAGE TABLES
  617 ;
  618 ;
  619 ;        SHELL PROCESS PAGE TABLES
  620 ;
  621 ;
  622
  623 P1PTBAS=.                                ;
  624
  625 .IF      NDF     PROCESS_PAGE_DEFINITIONS
  626          TMP=<SWP$C_SHLP1PT*512>         ; ALLOCATE SPACE FOR PAGE TABLES
  627          .BLKB   TMP                     ; ALLOCATE SPACE FOR PAGE TABLES
  628 .IFF
  629          .=.+<swp$c_shlp1pt*512>
  630 .ENDC
  631 SAV...=.                                 ; REMEMBER BASE OF PAGE TABLES
  632
  633
  634
  635 .IFTF                                    ; ALL ASSEMBLIES
  636
  637
  638
  639 ;
  640 ;        INITIALIZE VIRTUAL ADDRESS BASE
  641 ;
  642          VA...=1@31                      ;
  643
  644 ;        P1PTE   128*DBGPTCNT            ; DEBUGGER CONTEXT AREA
  645          VA...=VA...-<DBGPTCNT*128*512>  ; ACCOUNT FOR IT IN VA...
  646 DBGAREA=VA...                            ; ADDRESS OF DEBUG AREA
  647          P1PTE   1,URKW,K,PFIL           ; VECTOR PAGE
  648 VECTORS=VA...                            ;
  649
  650
  651
  652
  653 .IFF                                     ; EXCLUDE USRVECTOR MODULES
  654
  655
  656
  657
  658 DEFINE_CONSTANT CTL$GL_VECTORS,VECTORS   ; BASE OF VECTOR PAGE
  659          P1PTE   16                      ; PAGES FOR SYSTEM SERVICE VECTORS
  660 DEFINE_CONSTANT P1SYSVECTORS,VA...
  661 ;
  662 ;        THE FOLLOWING PAGE IS A USER-MODE WRITABLE CONTEXT PAGE
  663 ;        FOR MISC. USER-MODE PACKAGES THAT ARE MAINTAINED BY VMS.
  664 ;        (SEE DETAILED ALLOCATION OF PAGE BELOW)
  665 ;
  666          P1PTE   1,UW,K,DZRO             ; GLOBAL USER-MODE OWN STORAGE PAGE
  667 UWVECPAG=VA...
  668 ;
  669 ;        The following pages are for use by VMS kernel mode
  670 ;        facilities for uninitialized, process OWN storage. This page also
  671 ;        serves as the guard page for the supervisor stack.
  672 ;
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 14
X-45U3 SHELL PROCESS PAGE TABLES 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (5)

  673         P1PTE   2,URKW,K,DZRO           ; Kernel mode, demand zero data pages
  674 DEFINE_CONSTANT CTL$A_PRCPRM_KDATA,VA...,<PROCESS_SCHED,SHELL>
  675         P1PTE   SSTACK,URSW,S,DZRO,SSPINI       ; SUPERVISOR STACK
  676         P1PTE   ESTACK,SREW,E,DZRO,ESPINI       ; EXECUTIVE STACK
  677         P1PTE   KSTACK,SRKW,K,PFIL,KSPINI       ; KERNEL STACK
  678 DEFINE_CONSTANT CTL$GL_KSTKBAS,VA...,<PROCESS_SCHED,SHELL> ; MAXIMUM TOP OF KERNEL S
  679         P1PTE   KSTACK_EX,NA,K,PFIL             ; EXPANSION FOR  KERNEL STACK
  680 DEFINE_CONSTANT CTL$GL_KSTKBASEXP,VA...,<PROCESS_SCHED,SHELL> ; MAXIMUM TOP OF EXPAN
  681 DEFINE_CONSTANT CTL$GL_KSPINI,KSPINI,<PROCESS_SCHED,SHELL> ; INITIAL KERNEL STACK
  682         P1PTE   1                       ; NULL STOPPER PAGE
  683         P1PTE   KRP_COUNT,URKW,K,DZRO   ; P1 LOOKASIDE LIST FOR KERNEL MODE
  684 DEFINE_CONSTANT CTL$GL_KRP,VA...,<PROCESS_SCHED,SHELL>  ; ADDRESS USED BY PROCSTRT T
  685         P1PTE   1,URSW,E,DZRO           ; IMAGE HEADER BUFFER
  686 DEFINE_CONSTANT MMG$IMGHDRBUF,VA...,<IMAGE_ACTIVATOR,SHELL> ; ADDRESS OF IMAGE HEADE
  687         P1PTE   3,UREW,K,DZRO           ; VECTORS FOR USER SYS SRV & ERR MSGS
  688 DEFINE_CONSTANT CTL$A_DISPVEC,VA...,<IMAGE_ACTIVATOR,SHELL> ; ADR OF VECTOR PAGES FO
  689         P1PTE   4,UW,U,DZRO             ; DEBUGGER CONTEXT
  690         P1PTE   IMGACTBUF,UREW,E,DZRO   ; IMAGE ACTIVATOR SCRATCH PAGES (8)
  691 DEFINE_CONSTANT IAC$AL_IMGACTBUF,VA...,<IMAGE_ACTIVATOR,SHELL>  ; ADR OF IMAGE ACTIV
  692 CLIDATAEND=VA...
  693         P1PTE   12,URSW,S,DZRO          ; COMMMAND INTERPRETER DATA PAGES
  694 CLIDATAPAG=VA...                        ; BASE ADDRESS OF CLI DATA PAGE
  695         P1PTE   1,UREW,E,DZRO           ; IMAGE ACTIVATOR CONTEXT
  696 IMGACTCTX=VA...
  697         P1PTE   NSA$K_IDT_PAGES,KW,K,DZRO ; SECURITY AUDITING IMPURE DATA TABLE
  698 DEFINE_CONSTANT NSA$T_IDT,VA...,SECURITY
  699 ;
  700 ;       THE FOLLOWING PAGE IS A WRITABLE CONTEXT PAGE FOR
  701 ;       THE USE OF THE COMPATIBILITY MODE EMULATOR AND EXCEPTION
  702 ;
  703         P1PTE   2,UW,K,DZRO             ; WRITABLE PAGES FOR COMPATIBILITY MODE
  704 DEFINE_CONSTANT CTL$AG_CMEDATA,VA...,<PROCESS_SCHED,SHELL>
  705 ;
  706 ;       THE FOLLOWING PAGES ARE FOR USE BY BASIC/BASIC-PLUS TO PROVIDE
  707 ;       THE CORE COMMON REQUIRED TO PASS DATA WHEN CHAINING FROM IMAGE
  708 ;       TO IMAGE.  AN IDENTICAL AREA, NEGATIVELY DISPLACED HAS BEEN ALLOCATED
  709 ;       FOR USERS AND CSS.
  710 ;
  711         P1PTE   COMMON,UW,K,DZRO        ; DEC CORE COMMON PAGES
  712 DEFINE_CONSTANT CTL$A_COMMON,VA...,<PROCESS_SCHED,SHELL> ; BASE ADDRESS
  713         P1PTE   COMMON,UW,K,DZRO        ; USER CORE COMMON PAGES
  714
  715 DEFINE_CONSTANT PIO$A_RMS_PIOEND,VA...,<PROCESS_SCHED,FILES_VOLUMES,SHELL>
  716         P1PTE   1,UREW,E,DZRO           ; PROCESS IFB/IRB TABLES
  717 PIOTBL=VA...
  718         P1PTE   4,UREW,E,DZRO           ; RMS DIRECTORY CACHE PAGES
  719 DEFINE_CONSTANT PIO$A_DIRCACHE,VA...,<PROCESS_SCHED,FILES_VOLUMES,SHELL>
  720         P1PTE   1,UREW,E,DZRO           ; EXTENSION TO RMS POINTER PAGE
  721         P1PTE   1,UREW,E,PFIL           ; RMS POINTER PAGE
  722 DEFINE_CONSTANT PIO$A_RMS_PIOBASE,VA...,<PROCESS_SCHED,FILES_VOLUMES,SHELL>
  723
  724 PIOBASE=VA...                           ;
  725         P1PTE   2,UW,K,DZRO             ; VWS area
  726 DEFINE_CONSTANT CTL$A_VWS,VA...,<PROCESS_SCHED,SHELL>
  727 USPINI=VA...                            ; USER STACK AREA
  728
  729 END...=.                                ; END OF CONTROL REGION
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 15
X-45U3 SHELL PROCESS PAGE TABLES 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (5)

  730
  731 P1PTLEN=<1@31-VA...>@-9                 ; ENTRIES IN P1PT
  732
  733 ;
  734 ; Make sure there are enough system page table pages
  735 ; to cover SHELL P1 pages.
  736 ;
  737 .IF NDF PRMSW
  738 ASSUME  <<P1PTLEN+127>/128> EQ SWP$C_SHLP1PT
  739 .ENDC
  740
  741 DEFINE_CONSTANT SWP$C_SHLFPTE,<<.-P1PTBAS>@-2>,MEMORY_MANAGEMENT ; FREE P1PT IN BASI
  742 ;
  743 ; RESTORE LOCATION POINTER
  744 ;
  745         .=SAV...                        ; FINISHED WITH PAGE TABLES
  746         NOPFIL_CHARGE=KSTACK+1          ; No page file for kernel stack or vector pa
  747         PFILPGCNT=PFILPGCNT-NOPFIL_CHARGE
  748                                 ; DEFINE GLOBAL VALUE FOR SHELL PAGE FILE REQUIREMEN
  749         DEFINE_CONSTANT SWP$C_SHELLPFIL,PFILPGCNT,PROCESS_SCHED
  750
  751                                         ; Define global value for shell page
  752                                         ; file requirement for non-dzro pages
  753         DEFINE_CONSTANT SWP$C_SHELLPFIL_NDZRO,PFILPGCNT_NDZRO,PROCESS_SCHED
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 16
X-45U3 BODY OF SHELL PROCESS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

  755          .SBTTL  BODY OF SHELL PROCESS
  756 ;
  757 ;       BODY OF SHELL PROCESS
  758 ;
  759          .=SAV...                        ; POSITION TO END OF HEADER
  760 ;-----------------------------------------------------------------------
  761 ;
  762 ;       VECTOR PAGE
  763 ;               ***** NOTE: The cells in this page must not move.  There are
  764 ;               *****       facilities which locate cells via offsets from
  765 ;               *****       the top of this page.  If you delete a cell
  766 ;               *****       subsitute a .LONG 0 and comment it as SPARE.  If
  767 ;               *****       you add a cell either replace a SPARE cell or
  768 ;               *****       add it to the end.
  769 ;
  770 ;-----------------------------------------------------------------------
  771 DEFINE_CONSTANT SWP$AL_PTRPAG,.-SHELL,PROCESS_SCHED ; ADDRESS IN SWAPPER MAP
  772
  773
  774 .IFT                                    ; ONLY USRVECTOR MODULE
  775
  776
  777          .=VECTORS                       ; SHOW ACTUAL ADDRESSES IN LISTING
  778
  779
  780 .IFTF                                   ; ALL ASSEMBLIES
  781
  782
  783
  784 CTLVECPAG=.                             ; BASE OF VECTOR PAGE
  785          CTL     GW_NMIOCH,IO           ; NUMBER OF CHANNELS
  786          .WORD   0                      ; FILLED IN IN PROCSTRT
  787
  788          CTL     GW_CHINDX,IO           ; MAXIMUM CHANNEL INDEX
  789          .WORD   0                      ; FILLED IN IN PROCSTRT
  790
  791          CTL     GL_LNMHASH,LOGICAL_NAMES ; process Logical Name HASH table
  792          .LONG   0                      ;  pointer
  793          CTL     GL_LNMDIRECT,LOGICAL_NAMES ; process Logical Name DIRECTory
  794          .LONG   0                      ;  pointer
  795
  796 ;
  797 ;       ARRAY OF STACK POINTERS
  798 ;
  799 ;       THE -1 ENTRY OF CTL$AL_STACK IS USED TO DETERMINE THE MAXIMUM
  800 ;       EXTENT OF THE KERNEL STACK.
  801 ;
  802          .LONG   CTL$GL_KSTKBAS         ; MAXIMUM TOP OF KERNEL STACK
  803          CTL     AL_STACK,STABLE        ; STACK POINTER RESTART ARRAY
  804          .LONG   KSPINI                 ; KERNEL STACK POINTER
  805          .LONG   ESPINI                 ; EXEC STACK POINTER
  806          .LONG   SSPINI                 ; SUPERVISOR STACK POINTER
  807          .LONG   USPINI                 ; USER STACK POINTER
  808
  809 ;
  810 ; LOGICAL NAME TABLE TRANSLATION CACHE QUEUE HEADER
  811 ;
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 17
X-45U3 BODY OF SHELL PROCESS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

  812
  813          CTL     GQ_LNMTBLCACHE,LOGICAL_NAMES ; QUEUE HEADER FOR LOGICAL NAME TABLE
  814          .LONG   CTL$GQ_LNMTBLCACHE      ;  TRANSLATION CACHE (FORWARD LINK)
  815          .LONG   CTL$GQ_LNMTBLCACHE      ;                    (BACK LINK)
  816
  817 ;
  818 ; THE FOLLOWING THREE VECTORS MUST BE ADJACENT
  819 ;
  820
  821          CTL     GL_CMSUPR,STABLE        ; SUPERVISOR CHANGE MODE DISPATCHER
  822          .LONG   0                       ; NULL
  823
  824          CTL     GL_CMUSER,STABLE        ; USER CHANGE MODE DISPATCHER
  825          .LONG   0                       ; NULL
  826
  827          CTL     GL_CMHANDLR,STABLE      ; COMPATIBILITY MODE HANDLER
  828          .LONG   0
  829
  830          CTL     AQ_EXCVEC,STABLE        ; PRIMARY/SECONDARY EXCEPTION VECTORS
  831          .LONG   0                       ; KERNEL MODE PRIMARY
  832          .LONG   0                       ; KERNEL MODE SECONDARY
  833          .LONG   0                       ; EXEC MODE PRIMARY
  834          .LONG   0                       ; EXEC MODE SECONDARY
  835          .LONG   0                       ; SUPERVISOR MODE PRIMARY
  836          .LONG   0                       ; SUPERVISOR MODE SECONDARY
  837          .LONG   0                       ; USER MODE PRIMARY
  838          .LONG   0                       ; USER MODE SECONDARY
  839
  840          CTL     GL_THEXEC,STABLE        ; EXECUTIVE TERMINATION HANDLER
  841          .LONG   0                       ; NULL
  842
  843          CTL     GL_THSUPR,STABLE        ; SUPERVISOR TERMINATION HANDLER
  844          .LONG   0                       ; NULL
  845
  846 ;        CTL     GL_THUSER,STABLE        ; USER TERMINATION HANDLER
  847          .LONG   0                       ; NULL
  848
  849 ;        FORMER LOCATION OF COMPATIBILITY MODE CONTEXT
  850          CTL     GQ_COMMON               ; CORE COMMON DESCRIPTOR
  851          .LONG   512*COMMON              ; SIZE IN BYTES
  852          .LONG   CTL$A_COMMON            ; AND ADDRESS
  853
  854          CTL     GL_GETMSG,MISC          ; PER-PROCESS VECTOR TO USER MESSAGE
  855          .LONG   0                       ; DISPATCHER
  856
  857          CTL     AL_STACKLIM,STABLE      ; STACK LIMIT ARRAY (INDEXED BY MODE)
  858          .LONG   CTL$GL_KSTKBAS          ; KERNEL STACK LO (TOP) LIMIT
  859          .LONG   KSPINI                  ; EXEC   STACK LO (TOP) LIMIT
  860          .LONG   ESPINI                  ; SUPER  STACK LO (TOP) LIMIT
  861          .LONG   0                       ; USER STACK (NOT CHECKED)
  862
  863          CTL     GL_CTLBASVA,PROCESS_SCHED ; BASE CONTROL REGION ADDRESS
  864          .LONG   0                       ; FILLED IN BY PROCSTRT
  865
  866          CTL     GL_IMGHDRBF,IMAGE_ACTIVATOR ; ADDRESS OF IMAGE ACTIVATOR'S IMAGE
  867                                          ; HEADER BUFFER, IF IMAGE IS ACTIVE;
  868          .LONG   0                       ; 0 IF NO IMAGE ACTIVE
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 18
X-45U3 BODY OF SHELL PROCESS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

869
870         CTL     GL_IMGLSTPTR,-          ; ADDRESS OF ICB LIST (FOR DEBUGGER)
871                 <IMAGE_ACTIVATOR>
872         .LONG   IAC$GL_IMAGE_LIST
873
874         CTL     GL_PHD,PROCESS_SCHED    ; ADDRESS OF PHD WINDOW
875         .LONG   0                       ; INIT BY INSWAP
876
877         CTL     GQ_ALLOCREG,MEMORY_MANAGEMENT ; HEAD OF PROCESS ALLOCATION REGION PO
878         .LONG   0,0                     ;       (Filled in by PROCSTRT)
879
880         CTL     GQ_MOUNTLST,-           ; MOUNTED DEVICE LIST
881                 <FILES_VOLUMES>
882         .LONG   CTL$GQ_MOUNTLST         ; FORWARD LINK
883         .LONG   CTL$GQ_MOUNTLST         ; BACKWARD LINK ( LIST EMPTY )
884
885 ;****************************************************************************
886 ; NOTE: The order of the accounting data between CTL$T_USERNAME and
887 ;       CTL$T_REMOTEID should be preserved!
888 ;****************************************************************************
889
890         CTL     T_USERNAME,STABLE       ; USERNAME
891         .BLKB   JIB$S_USERNAME          ;  Filled in by PROCSTRT
892
893         CTL     T_ACCOUNT,STABLE        ; ACCOUNT NAME
894         .BLKB   JIB$S_ACCOUNT           ;  Filled in by PROCSTRT
895
896 .IF NDF PRMSW
897 ASSUME <CTL$T_ACCOUNT - CTL$T_USERNAME> EQ <JIB$T_ACCOUNT - JIB$T_USERNAME>
898 .ENDC
899
900         CTL     GQ_LOGIN,STABLE         ; SYSTEM TIME AT PROCESS CREATION
901         .LONG   0,0                     ;
902
903         CTL     GL_FINALSTS,STABLE      ; FINAL EXIT STATUS FOR PROCESS
904         .LONG   0                       ;
905
906         CTL     GL_WSPEAK,COUNTERS      ; PEAK WORKING SET SIZE
907         .LONG   0                       ;
908
909         CTL     GL_VIRTPEAK,COUNTERS    ; PEAK VIRTUAL SIZE
910         .LONG   0                       ;
911
912         CTL     GL_VOLUMES,COUNTERS     ; COUNT OF VOLUMES MOUNTED
913         .LONG   0                       ;
914
915         CTL     GQ_ISTART,COUNTERS      ; IMAGE ACTIVATION TIME
916         .LONG   0,0
917
918         CTL     GL_ICPUTIM,COUNTERS     ; INITIAL IMAGE CPU TIME
919         .LONG   0
920
921         CTL     GL_IFAULTS,COUNTERS     ; INITIAL IMAGE FAULT COUNT
922         .LONG   0
923
924         CTL     GL_IFAULTIO,COUNTERS    ; INITIAL IMAGE FAULT I/O COUNT
925         .LONG   0
```

```
926
927         CTL     GL_IWSPEAK,COUNTERS     ; IMAGE WORKING SET PEAK
928         .LONG   0
929
930         CTL     GL_IPAGEFL,COUNTERS     ; IMAGE PAGE FILE PEAK USAGE
931         .LONG   0
932
933         CTL     GL_IDIOCNT,COUNTERS     ; INITIAL IMAGE DIRECT I/O COUNT
934         .LONG   0
935
936         CTL     GL_IBIOCNT,COUNTERS     ; INITIAL IMAGE BUFFERED I/O COUNT
937         .LONG   0
938
939         CTL     GL_IVOLUMES,COUNTERS    ; INITIAL IMAGE VOLUME MOUNT COUNT
940         .LONG   0
941
942         CTL     T_NODEADDR,NETWORKS     ; REMOTE NODE ADDRESS (BINARY)
943         .BYTE   0                               ; BYTE COUNT BYTE
944         .BLKB   6                               ; 6 BYTES MAX
945
946         CTL     T_NODENAME,NETWORKS     ; REMOTE NODE NAME (ASCII)
947         .BYTE   0                               ; BYTE COUNT BYTE
948         .BLKB   6                               ; 6 CHARACTERS MAX
949
950         CTL     T_REMOTEID,NETWORKS     ; REMOTE ID
951         .BYTE   0                               ; BYTE COUNT BYTE
952         .BLKB   16                              ; 16 CHARACTERS MAX
953
954         .BLKB   1                       ; LONGWORD ALIGN SECTION
955
956 ;*********************************
957 ; End of adjacent accounting data
958 ;*********************************
959
960         CTL     GQ_PROCPRIV,STABLE      ; PROCESS PRIVILEGE MASK
961         .LONG   -1,-1                   ;  ALL PRIVILEGES PERMITTED
962
963         CTL     GL_USRCHMK,STABLE       ; PER-PROCESS VECTOR TO USER CHANGE MODE
964         .LONG   0                       ; TO KERNEL HANDLER
965
966         CTL     GL_USRCHME,STABLE       ; PER-PROCESS VECTOR TO USER CHANGE MODE
967         .LONG   0                       ; TO EXECUTIVE HANDLER
968
969         CTL     GL_POWERAST,STABLE      ; POWER FAIL AST ADDRESS
970         .LONG   0                       ;
971         CTL     GB_PWRMODE,STABLE       ; ACCESS MODE FOR POWER FAIL AST
972         .BYTE   0                       ;
973
974         CTL     GB_SSFILTER,STABLE      ; SYS SERV INHIBIT FILTER MASK
975         .BYTE   0
976
977         CTL     GB_REENABLE_ASTS,VOLATILE ; USER MODE THREAD MUST CALL $SETAST(1)
978         .BLKB   1
979
980         .BLKB   1                       ; SPARE
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 20
X-45U3 BODY OF SHELL PROCESS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

  982
  983 ;
  984 ;       ARRAY OF FINAL EXCEPTION VECTORS - ONE PER MODE
  985 ;
  986 ;       THESE VECTORS ARE USED TO LOCATE AN EXCEPTION HANDLER WHEN ALL
  987 ;       OTHER METHODS FAIL.  PERMITS DEBUGGERS TO RECEIVE CONTROL EVEN
  988 ;       WHEN THE STACK IS CLOBBERED.
  989 ;
  990         CTL     AL_FINALEXC,STABLE      ; FINAL EXCEPTION HANDLER ARRAY
  991         .LONG   EXE$EXCPTN              ; KERNEL MODE
  992         .LONG   EXE$EXCPTNE             ; EXECUTIVE MODE
  993         .LONG   0                       ; SUPERVISOR MODE
  994         .LONG   0                       ; USER MODE
  995 ;
  996 ;       POINTER TO BASE OF CHANNELS
  997 ;
  998         CTL     GL_CCBBASE,IO           ; BASE OF I/O CHANNELS
  999         .LONG   0                       ; FILLED IN IN PROCSTRT
 1000         CTL     GQ_DBGAREA,MISC         ; DEBUG AREA
 1001         .LONG   64*1024                 ; SIZE
 1002         .LONG   DBGAREA                 ; ADDRESS
 1003 ;
 1004 ;  POINTER TO RMS IMAGE
 1005 ;
 1006         CTL     GL_RMSBASE,FILES_VOLUMES
 1007         .LONG   0                       ; POINT TO RMS IN SYSTEM SPACE
 1008 ;
 1009 ;       PROCESS PERMANENT MESSAGE SECTION POINTER AND DEFAULT DISPLAY FLAGS
 1010 ;
 1011         CTL     GL_PPMSG,MISC           ; ADDRESS OF PROCESS PERM. MSG SECTION
 1012         .QUAD   0                       ; STARTING/ENDING ADDRESS OF SECTION
 1013
 1014         CTL     GB_MSGMASK,MISC         ; DEFAULT MESSAGE DISPLAY FLAGS
 1015         .BYTE   15                      ; DEFAULT: FAC,SEV,IDENT AND TEXT (ALL)
 1016
 1017         CTL     GB_DEFLANG,MISC         ; DEFAULT MESSAGE LANGUAGE
 1018         .BYTE   0                       ; (CURRENTLY UNUSED)
 1019
 1020         CTL     GW_PPMSGCHN,MISC        ; CHANNEL TO PROCESS PERM. MESSAGE
 1021         .WORD   0                       ; SECTION (MAPPED IN CTL$GL_PPMSG)
 1022
 1023         CTL     GL_USRUNDWN,MISC        ; PER-PROCESS VECTOR TO USER RUNDOWN
 1024         .LONG   0                       ;  SERVICE
 1025
 1026         CTL     GL_PCB,PROCESS_SCHED    ; ADDRESS OF PROCESS CONTROL BLOCK
 1027         .LONG   0                       ; INIT BY PROCSTRT
 1028
 1029         CTL     GL_RUF,FILES_VOLUMES    ; POINTER TO RECOVERY UNIT BLOCKS
 1030         .LONG   0
 1031
 1032         CTL     GL_SITESPEC,MISC        ; SITE-SPECIFIC PER-PROCESS CELL
 1033         .LONG   0
 1034
 1035         CTL     GL_KNOWNFIL,FILES_VOLUMES ; PROCESS KNOWN FILE LIST POINTER
 1036         .LONG   0
 1037
 1038         CTL     AL_IPASTVEC,MISC        ; VECTOR OF IPAST ADDRESSES
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 21
X-45U3 BODY OF SHELL PROCESS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

 1039          .LONG   0,0,0,0,0,0,0,0

 1040
 1041          CTL     GL_CMCNTX,MISC          ; ADDRESS OF AME CONTEXT PAGE
 1042          .LONG   CTL$AL_CMCNTX
 1043
 1044          CTL     GL_IAFLNKPTR,IMAGE_ACTIVATOR ; ADDRESS OF IAF LIST (FOR DEBUGGER)
 1045          .LONG   CTL$GL_IAFLINK
 1046
 1047          CTL     GL_F11BXQP,FILES_VOLUMES ; ADDRESS OF F11B XQP QUEUE AND
 1048          .LONG   0                       ; DISPATCH VECTORS
 1049
 1050          CTL     GQ_POALLOC,MISC         ; HEADER OF P0 EXTENTION TO PROCESS
 1051          .LONG   0,0                     ; ALLOCATION REGION
 1052
 1053          CTL     GL_PRCALLCNT,MISC       ; COUNT OF BYTES OF PROCESS ALLOCATION
 1054          .LONG   0                       ; REGION USABLE BY IMAGE REQUESTS.
 1055
 1056          CTL     GL_RDIPTR,SECURITY      ; POINTER TO RIGHTS DATABASE IDENTIFIER (RDI
 1057          .LONG   0
 1058
 1059          CTL     GL_LNMDIRSEQ,LOGICAL_NAMES ; SEQUENCE NUMBER FOR CACHE OF LOGICAL
 1060          .LONG   0                       ;  NAME TABLE TRANSLATIONS
 1061
 1062          CTL     GQ_HELPFLAGS,MISC       ; HELP FLAGS, ONE LONGWORD FOR USE
 1063          .LONG   0,0                     ; NOW, ONE FOR LATER
 1064
 1065          CTL     GQ_TERMCHAR,IO          ; HOME FOR TERMINAL CHARACTERISTICS
 1066          .LONG   0 , 0                   ; NEED A QUADWORD
 1067
 1068          CTL             GL_KRPFL,MISC   ; P1 POOL LOOKASIDE LIST FORWARD LINK
 1069          .LONG           CTL$GL_KRPFL
 1070          CTL             GL_KRPBL,MISC   ; P1 POOL LOOKASIDE LIST BACKWARD LINK
 1071          .LONG           CTL$GL_KRPFL
 1072
 1073          CTL             GL_CREPRC_FLAGS,PROCESS_SCHED ; $CREPRC FLAGS
 1074          .LONG   0                       ;
 1075
 1076          CTL     GL_THCOUNT,STABLE       ; COUNT OF TERMINATION HANDLERS FOR
 1077          .LONG   0                       ;  EXEC MODE,
 1078          .LONG   0                       ;  SUPERVISOR MODE, AND
 1079          .LONG   0                       ;  USER MODE
 1080
 1081          CTL     GQ_CWPS_Q1,VOLATILE     ; CWPS temporary
 1082          .LONG   CTL$GQ_CWPS_Q1
 1083          .LONG   CTL$GQ_CWPS_Q1
 1084
 1085          CTL     GQ_CWPS_Q2,VOLATILE     ; CWPS temporary
 1086          .LONG   CTL$GQ_CWPS_Q2
 1087          .LONG   CTL$GQ_CWPS_Q2
 1088
 1089          CTL     GL_CWPS_L1,VOLATILE     ; CWPS temporary
 1090          .LONG   0
 1091          CTL     GL_CWPS_L2,VOLATILE     ; CWPS temporary
 1092          .LONG   0
 1093          CTL     GL_CWPS_L3,VOLATILE     ; CWPS temporary
 1094          .LONG   0
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 22
X-45U3 BODY OF SHELL PROCESS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

 1095          CTL     GL_CWPS_L4,VOLATILE     ; CWPS temporary
 1096          .LONG   0
 1097
 1098          CTL     GQ_XCB_QUE,VOLATILE     ; Process list of active transactions
 1099          .LONG   CTL$GQ_XCB_QUE
 1100          .LONG   CTL$GQ_XCB_QUE
 1101
 1102          CTL     GL_PRCPRM_KDATA2,STABLE ; Address of kernel data extension area
 1103          .LONG   0
 1104
 1105          CTL     GL_USRUNDWN_EXEC,VOLATILE ; Per-process vector to user rundown
 1106          .LONG   0                       ; service (EXEC mode)
 1107
 1108 ;****************************************************************************
 1109 ;
 1110 ;       END OF VECTOR PAGE
 1111 ;
 1112 ;****************************************************************************
 1113 CTLVECEND=.
 1114          .IF GREATER     <CTLVECEND-CTLVECPAG>-512
 1115          .ERROR          ; *** VECTOR PAGE NOW LARGER THAN A PAGE ***
 1116          .ENDC
 1117
 1118
 1119 .IFF                                    ; EXCLUDE USRVECTOR MODULE
 1120
 1121
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 23
X-45U3 PROCESS I/O SEGMENT 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

 1123          .SBTTL  PROCESS I/O SEGMENT
 1124 ;----------------------------------------------------------------------------

 1125 ;
 1126 ;       PROCESS I/O SEGMENT INITIAL CONTENT
 1127 ;
 1128 ;----------------------------------------------------------------------------

 1129 .IF      NDF     PRMSW
 1130 .IF      NDF     PROCESS_PAGE_DEFINITIONS
 1131          .ALIGN  PAGE                    ; PAGE BOUNDARY
 1132 .IFF
 1133          .=<.+511>&^XFE00
 1134 .ENDC
 1135 .ENDC
 1136
 1137 PIO:                                     ; BASE OF PROCESS I/O SEGMENT
 1138
 1139          PIO     GL_FMLH,FILES_VOLUMES   ; FREE MEMORY LIST HEAD
 1140          .LONG   PIO$GL_FMLH,PIO$GL_FMLH ;       (Empty List)
 1141
 1142          PIO     GL_IIOFSPLH,FILES_VOLUMES ; FREE LIST HEADER FOR IMAGE I/O SEG
 1143          .LONG   PIO$GL_IIOFSPLH         ;
 1144          .LONG   PIO$GL_IIOFSPLH         ;
 1145
 1146          PIO     GW_STATUS,STABLE        ; RMS OVERALL STATUS
 1147          .WORD   0                       ;
 1148
 1149
 1150 DEFINE_CONSTANT PIO$S_EODSTR,16,<FILES_VOLUMES> ; SIZE OF STRING + COUNT BYTE
 1151          PIO     GT_ENDSTR,FILES_VOLUMES ; END OF DATA STRING
 1152          .BLKB   PIO$S_EODSTR            ; (COUNTED STRING)
 1153
 1154          PIO     GW_DFPROT,FILES_VOLUMES ; DEFAULT FILE PROTECTION
 1155          .WORD   ^XFA00                  ; SYS:RWED, OWN:RWED, GROUP: RE, WORLD:N
 1156
 1157          PIO     GB_DFMBC,FILES_VOLUMES  ; DEFAULT MULTI-BLOCK COUNT
 1158          .BLKB   1
 1159
 1160          PIO     GB_DFMBFSDK,FILES_VOLUMES ; DEFAULT MULTI-BUFFER COUNT SEQ. DISK
 1161          .BLKB   1
 1162
 1163          PIO     GB_DFMBFSMT,FILES_VOLUMES ; DEFAULT MULTI-BUFFER COUNT MAGTAPE
 1164          .BLKB   1
 1165
 1166          PIO     GB_DFMBFSUR,FILES_VOLUMES ; DEFAULT MULTI-BUFFER COUNT UNIT REC.
 1167          .BLKB   1
 1168
 1169          PIO     GB_DFMBFREL,FILES_VOLUMES ; DEFAULT MULTI-BUFFER COUNT RELATIVE
 1170          .BLKB   1
 1171
 1172          PIO     GB_DFMBFIDX,FILES_VOLUMES ; DEFAULT MULTI-BUFFER COUNT INDEXED
 1173          .BLKB   1
 1174
 1175          PIO     GB_DFMBFHSH,FILES_VOLUMES ; DEFAULT MULTI-BUFFER COUNT HASHED
 1176          .BLKB   1
 1177
 1178          PIO     GB_DFNBC,FILES_VOLUMES  ; Network block count transfer size
 1179          .BLKB   1
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 24
X-45U3 PROCESS I/O SEGMENT 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

 1180
 1181          PIO     GB_RMSPROLOG,FILES_VOLUMES ; Structure level for RMS files
 1182          .BLKB   1
 1183
 1184          PIO     GW_RMSEXTEND,FILES_VOLUMES ; Extend quantity for RMS files
 1185          .BLKW   1
 1186
 1187          PIO     GB_JNL_STALL_CNT,FILES_VOLUMES ; Count of journal stalled threads
 1188          .BLKB   1
 1189
 1190 .IF      NDF     PRMSW
 1191 .IF      NDF     PROCESS_PAGE_DEFINITIONS
 1192          .ALIGN  LONG
 1193 .IFF
 1194          .=<.+3>&^XFFFC
 1195 .ENDC
 1196 .ENDC
 1197          PIO     GL_DIRCACHE,FILES_VOLUMES ; DIRECTORY CACHE LIST HEAD
 1198          .LONG   PIO$GL_DIRCACHE         ; EMPTY LIST
 1199          .LONG   PIO$GL_DIRCACHE
 1200
 1201          PIO     GL_DIRCFRLH,FILES_VOLUMES ; FREE LIST FOR DIRECTORY CACHE NODES
 1202          .LONG   0                       ;   (NOTE: SINGLY LINKED)
 1203
 1204          PIO     GQ_RUB_LH,FILES_VOLUMES    ; RMS Recovery Unit Block (RUB) list head
 1205          PIO     GL_RUB_FLINK,FILES_VOLUMES ;   RUB queue forward link
 1206          .LONG   PIO$GQ_RUB_LH
 1207          PIO     GL_RUB_BLINK,FILES_VOLUMES ;   RUB queue backward link
 1208          .LONG   PIO$GQ_RUB_LH
 1209
 1210          PIO     GL_NXTIRBSEQ,FILES_VOLUMES ; Next sequence number for IRB$L_IDENT
 1211          .LONG   0
 1212
```

```
1214
1215 .IF     NDF     PRMSW
1216 .IF     NDF     PROCESS_PAGE_DEFINITIONS
1217         .ALIGN  QUAD
1218 .IFF
1219         .=<.+7>&^XFFF8
1220 .ENDC
1221 .ENDC
1222 ;
1223 ;       PROCESS I/O SEGMENT CONTEXT AREA
1224 ;
1225         PIO     GW_PIOIMPA,FILES_VOLUMES
1226         .WORD   0                       ; FLAGS
1227         .WORD   PRT$C_UREW              ; I/O BUFFER PROTECTION
1228         .LONG   0,0                     ; PIO SEGMENT, SET UP BY PROCSTRT
1229
1230         PIOL    A1                      ; FREE PAGE LIST HEAD
1231         .LONG   PIOLA1,PIOLA1           ; FREE LIST HEADER
1232         .BLKL   1                       ; SP SAVED LONGWORD
1233         .LONG   PIO2A10                 ; IFAB TABLE ADDRESS
1234         .LONG   PIO2A20                 ; IRAB TABLE ADDRESS
1235         .LONG   IMP$C_NPIOFILES         ; # OF SLOTS PER TABLE
1236
```

```
1238 ;
1239 ;       IMAGE I/O SEGMENT CONTEXT AREA
1240 ;
1241 .IF     NDF     PRMSW
1242 .IF     NDF     PROCESS_PAGE_DEFINITIONS
1243         .ALIGN  QUAD
1244 .IFF
1245         .=<.+7>&^XFFF8
1246 .ENDC
1247 .ENDC
1248         PIO     GW_IIOIMPA,FILES_VOLUMES ; IMAGE I/O IMPURE AREA
1249         .WORD   1                        ; FLAGS WITH IIO SEGMENT SET
1250         .WORD   PRT$C_UREW               ; PROTECTION STO SET ON PAGES
1251
1252         .LONG   0                        ; MUST BE INITIALIZED BY THE IMAGE
1253                                          ; ACTIVATOR TO THE ADDRESS OF THE FIRST
1254                                          ; PAGE IN THE IMAGE I/O SEGMENT.
1255                                          ; (NOTE: THIS LONGWORD IS REFERENCED BY:
1256                                          ; PIO$GW_IIOIMPA+IMP$L_IOSEGADDR)
1257         .LONG   0                        ; LENGTH OF IMAGE I/O SEGMENT IN BYTES
1258                                          ; MUST BE INITIALIZED BY THE IMAGE ACTIVATOR
1259                                          ; (NOTE: THIS LONGWORD IS REFERENCED BY:
1260                                          ; PIO$GW_IIOMPA+IMP$L_IOSEGLEN)
1261
1262         PIOL    B1                       ;
1263         .LONG   PIOLB1,PIOLB1            ; FREEPAGE LIST HEAD
1264
1265
1266         PIOL    B2                       ;
1267         .BLKL   1                        ; SP SAVE LONGWORD
1268
1269         .LONG   PIOLB10                  ; IFAB TABLE ADDRESS
1270         .LONG   PIOLB20                  ; IRAB TABLE ADDRESS
1271         .LONG   IMP$C_ENTPERSEG          ; # OF SLOTS PER TABLE
1272
1273         PIOL    B10                      ;
1274         .LONG   0                        ; LINK TO NEXT TABLE SEGMENT
1275         .BLKL   IMP$C_ENTPERSEG          ; IFAB TABLE SLOTS
1276
1277         PIOL    B20                      ;
1278         .LONG   0                        ; LINK TO NEXT TABLE SEGMENT
1279         .BLKL   IMP$C_ENTPERSEG          ; IRAB TABLE SLOTS
1280
1281         PIO     AL_RMSEXH,FILES_VOLUMES ; EXIT HANDLER CONTROL BLOCK
1282         .BLKL   2                        ;
1283         .LONG   1                        ; ARGUMENT COUNT
1284         .LONG   PIO$AL_RMSEXH+8          ; STORE EXIT CODE OVER ARGUMENT COUNT
1285
1286         PIO     GQ_IIODEFAULT,FILES_VOLUMES ; DEFAULT IMAGE I/O AREA
1287         .LONG   0,0
1288
1289         PIO     GL_LNKCSHADR,FILES_VOLUMES ; Logical link cache entry listhead
1290         .LONG   0
1291
1292         PIO     GL_RM0_RM_ID,FILES_VOLUMES     ; RMS/DDTM resource manager ID
1293         PIO     GL_RU_HANDLER_ID,FILES_VOLUMES ; RMS/RUF recovery unit handler ID
1294         .LONG   0                              ;    (mutually exclusive usage)
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 27
X-45U3 PROCESS I/O SEGMENT 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

1295
1296          PIO     GL_RU_FAILURE_COUNT,FILES_VOLUMES  ; Recovery unit failure count
1297          .LONG   0
1298
1299          PIO     GL_RU_WAIT_Q_FLINK,FILES_VOLUMES ; Recovery unit wait queue flink
1300          .LONG   0
1301
1302          PIO     GL_RU_WAIT_Q_BLINK,FILES_VOLUMES ; Recovery unit wait queue blink
1303          .LONG   0
1304
1305          PIO     GQ_NTRUB_LH,FILES_VOLUMES  ; RMS Network Recovery Unit Block (NTRUB)
1306          .LONG   PIO$GQ_NTRUB_LH            ;   NTRUB queue forward link
1307          .LONG   PIO$GQ_NTRUB_LH            ;   NTRUB queue backward link
1308
1309          PIO     GL_NTO_RM_ID,FILES_VOLUMES ; RMS/DDTM Network resource manager ID
1310          .LONG   0
1311
1312          PIO     GL_RESERVED0,FILES_VOLUMES ; Reserved for future use by RMS
1313          .LONG   0
1314
1315          PIO     GQ_RUF_TSB_LH,FILES_VOLUMES     ; RUF Transaction Stack Block (TSB)
1316          .LONG   PIO$GQ_RUF_TSB_LH               ;   RUF_TSB queue forward link
1317          .LONG   PIO$GQ_RUF_TSB_LH               ;   RUF_TSB queue backward link
1318
1319          PIO     GL_RESERVED1,FILES_VOLUMES ; Reserved for future use by RMS
1320          .LONG   0
1321          PIO     GL_RESERVED2,FILES_VOLUMES ; Reserved for future use by RMS
1322          .LONG   0
1323          PIO     GL_RESERVED3,FILES_VOLUMES ; Reserved for future use by RMS
1324          .LONG   0
1325          PIO     GL_RESERVED4,FILES_VOLUMES ; Reserved for future use by RMS
1326          .LONG   0
1327          PIO     GL_RESERVED5,FILES_VOLUMES ; Reserved for future use by RMS
1328          .LONG   0
1329          PIO     GL_RESERVED6,FILES_VOLUMES ; Reserved for future use by RMS
1330          .LONG   0
1331
1332
1333 .IF      NDF     PRMSW
1334 .IF      NDF     PROCESS_PAGE_DEFINITIONS
1335          .ALIGN  LONG
1336 .IFF
1337          .=<.+3>&^XFFFC
1338 .ENDC
1339 .ENDC
1340 ;
1341 ;        DEFAULT DIRECTORY INFORMATION
1342 ;
1343          PIO     GT_DDSTRING,FILES_VOLUMES       ; DEFAULT DIRECTORY STRING
1344
1345 DEFINE_CONSTANT FIL$GT_DDSTRING,PIO$GT_DDSTRING ; FILE READ DEFAULT DIRECTORY STRING
1346
1347 ; THE DEFAULT DIRECTORY STRING IN THE PROCESS QUOTA BLOCK MOVED FROM A
1348 ; CREATOR PROCESS TO A NEWLY CREATED PROCESS MUST BE AT LEAST AS LARGE
1349 ; AS THE LARGEST DIRECTORY STRING ALLOWED BY RMS.
1350
1351          ASSUME  PQB$S_DDSTRING GE FWA$C_MAXDIRLEN
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 28
X-45U3 PROCESS I/O SEGMENT 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

 1352
 1353 TEMP$...=.
 1354         .ASCIC  \[SYSEXE]\              ; DEFAULT DIRECTORY STRING
 1355
 1356 ; ****************************** NOTE WELL ******************************
 1357 ;
 1358 ; THE DEFAULT DIRECTORY STRING MUST BE THE LAST ELEMENT TO APPEAR IN THE
 1359 ; RMS CONTEXT AREA. IN FACT, IT HANGS OFF THE END OF THE PAGE. IF STORAGE
 1360 ; WERE ALLOCATED, THE SHELL WOULD GROW IN SIZE BY ONE PAGE, AN UNNECESSARY
 1361 ; WASTE OF SPACE. THE .BLKB DIRECTIVE HERE IS SYMBOLIC, TO EXPRESS THE SIZE
 1362 ; OF THE DIRECTORY STRING IF SPACE WERE REALLY ALLOCATED.
 1363 ;
 1364 ;        .BLKB   PQB$S_DDSTRING-<.-TEMP$...>     ; FILL TO MAXIMUM POSSIBLE SIZE
 1365 ;
 1366 ; ****************************** END OF NOTE ******************************
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 29
X-45U3 INITIALZE SHELL WITH SYSGEN PARAMETERS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELI

 1368          .SBTTL  INITIALZE SHELL WITH SYSGEN PARAMETERS
 1369 ;++
 1370 ; FUNCTIONAL DESCRIPTION:
 1371 ;       SWP$SHELINIT IS CALLED BY THE SWAPPER FOLLOWING THE INSWAP OF A SHELL
 1372 ;       PROCESS TO APPLY THE SYSGEN PARAMETERS AND CONFIGURE THE PROCESS
 1373 ;       FOR THE PROPER WORKING SET SIZE AND VIRTUAL ADDRESS SPACE.
 1374 ;
 1375 ; CALLING SEQUENCE:
 1376 ;       JSB     @#SWP$SHELINIT
 1377 ;
 1378 ; INPUT PARAMETERS:
 1379 ;       R4 - PCB ADDRESS
 1380 ;       R9 - SWAPPER MAP POINTER
 1381 ;       R10 - PTC$M_VALID!PTE$C_ERKW!PTE$M_MODIFY
 1382 ;       R11 - SWAPPER END MAP POINTER
 1383 ;
 1384 ; OUTPUT PARAMETERS:
 1385 ;       PHD AND CONTROL REGION FOR THE NEW PROCESS
 1386 ;
 1387 ;--
 1388
 1389 .IF     NDF     PRMSW
 1390 .IF     NDF     PROCESS_PAGE_DEFINITIONS
 1391          .ALIGN  PAGE                    ; START OUT ON PAGE BOUNDARY
 1392 .IFF
 1393          .=<.+511>&^XFE00
 1394 .ENDC
 1395 .ENDC
 1396
 1397 DEFINE_CONSTANT SWP$SHELINIT,.-SHELL
 1398          MOVL    PCB$L_PHD(R4),R5        ; GET PROCESS HEADER BASE ADDRESS
 1399          MOVL    G^SWP$GL_SHELLSIZ,R0    ; GET PAGES ALLOCATED FOR SHELL
 1400          MOVAL   (R9)[R0],R11            ; COMPUTE END ADDRESS IN MAP
 1401          ASHL    #2,#SWP$C_SHELLSIZ,R0   ; GET I/O SIZE OF SHELL
 1402          SUBL3   R9,R11,R1               ; COMPUTE SIZE OF EXTENSION PAGES
 1403          SUBL    R0,R1                   ; LESS I/O SIZE
 1404          ROTL    #3,R1,R1                ; CONVERT TO DOUBLE QUAD COUNT
 1405          ROTL    #9,#SWP$C_SHELLSIZ,R0   ; COMPUTE ADDRESS AT END OF I/O TRANSFER
 1406 10$:     CLRQ    (R0)+                   ; CLEAR PAGES
 1407          CLRQ    (R0)+                   ; NOT READ FROM SHELL
 1408          SOBGTR  R1,10$                  ;
 1409          MOVL    R5,R2                   ; VA OF PHD
 1410          JSB     G^MMG$SVAPTECHK         ; GET SVA OF FIRST PHD PTE
 1411          MOVL    (R9)+,(R3)+             ; MAP PROCESS HEADER FIXED PAGE
 1412          MOVZWL  G^SWP$GW_WSLPTE,R1      ; GET COUNT OF ADDED PAGES FOR WSL+PST
 1413          BEQL    30$                     ; BR IF NONE
 1414          DECL    R1                      ; LESS FIXED HEADER PAGE
 1415 20$:     MOVL    -(R11),(R3)+            ; MAP A WSL PAGE
 1416          SOBGTR  R1,20$                  ; DO THEM ALL
 1417 30$:     MOVZWL  G^SWP$GW_EMPTPTE,R0     ; GET COUNT OF EMPTY PAGES
 1418          MOVAL   (R3)[R0],R3             ; UPDATE MAP POINTER
 1419          MOVZWL  G^SWP$GW_BAKPTE,R0      ; GET COUNT OF BAK/WSL/VAL/LCK PTE
 1420 40$:     MOVL    -(R11),(R3)+            ; MAP BAK/WSL/VAL/LCK PAGES
 1421          SOBGTR  R0,40$                  ;
 1422          INVALIDATE_TB MODE=<@#>         ; INVALIDATE TRANSLATION BUFFER
 1423 ;
 1424 ;       ALL OF THE HEADER PAGES LESS PAGE TABLES HAVE NOW BEEN MAPPED
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 30
X-45U3 INITIALZE SHELL WITH SYSGEN PARAMETERS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELI

 1425 ;
 1426         MOVZWL  G^SWP$GW_IBALSETX,R8     ; GET BALANCE SLOT INDEX
 1427         MOVW    R8,PHD$W_PHVINDEX(R5)    ; SET INTO PROCESS HEADER
 1428         CVTWB   G^SGN$GW_DFPFC,PHD$B_DFPFC(R5)  ; SET DEFAULT PAGE FAULT CLUSTER
 1429 ;
 1430 ; Assign initial process pagefile and reserve SHELL pages.
 1431 ; Pass requested page file (if any) as a PREFERENCE ONLY.
 1432 ;
 1433         ASSUME  <PHD$S_PRCPGFLREFS/4> - ; Insure that page file accounting
 1434                 EQ <1@PTE$S_PRCPGFLX>   ; works correctly
 1435
 1436         MOVAL   PHD$L_PRCPGFLREFS(R5),R0; Init process page file ref counts
 1437         MOVL    #<1@PTE$S_PGFLVBN>,R1
 1438         .REPT   <PHD$S_PRCPGFLREFS/4>
 1439         MOVL    R1,(R0)+
 1440         .ENDR
 1441
 1442         MOVZBL  PCB$B_PGFLINDEX(R4),R1  ; Get requested SYSTEM page file index
 1443         JSB     G^MMG$ASNPRCPGFLP       ; Assign process page file
 1444
 1445         MOVZWL  G^SWP$GW_WSLPTE,R2      ;    WSL PTE pages
 1446         ADDL2   #SWP$C_SHELLPFIL_NDZRO,R2; + SHELL page file pages (not DZRO)
 1447         MOVZWL  G^SWP$GW_BAKPTE,R1
 1448         ADDL2   R1,R2                   ;  + BAK PTE pages
 1449         MOVZBL  G^SWP$GB_SHLP1PT,R1
 1450         ADDL2   R1,R2                   ;  + SHELL P1 page table pages
 1451         ADDL3   #127,R2,R1              ; Round up to multiple of 128
 1452         BICL2   #127,R1
 1453         JSB     G^MMG$RSRVPRCPGFL2      ; Unconditionally reserve rounded pages
 1454         SUBW2   R2,PHD$W_PRCPGFLPAGES(R5); Show actual pages as assigned
 1455         MOVW    PHD$W_PRCPGFLPAGES(R5),-; and propogate into total ref count
 1456                 PHD$W_PRCPGFLOPAGES(R5)
 1457         SUBL2   R2,PHD$L_PRCPGFLREFS(R5);[Rx=0] (implicitly process PF #0)
 1458         ADDL3   #SWP$C_SHELLPFIL,-      ; R1 = Total mapped pages with PF BS
 1459                 G^SGN$GL_PTPAGCNT,R1    ; (SHELL P1 + process page table pages)
 1460         SUBL2   R1,PHD$L_PPGFLVA(R5)    ; Account for mapped pages
 1461
 1462         MOVB    G^SGN$GB_PGTBPFC,PHD$B_PGTBPFC(R5); SET SYSTEM DEFAULT PT CLUSTER
 1463         MOVZWL  #SWP$C_KSTACK_WSL,PHD$L_WSLIST(R5); INIT POINTER TO LIST BASE
 1464         MOVL    G^SGN$GL_PHDPAGCT,R7    ; GET TOTAL COUNT OF HEADER PAGES
 1465         MOVZWL  G^SWP$GW_WSLPTE,R6      ; GET COUNT OF WSL HEADER PAGES
 1466         ASHL    #9,R6,PHD$L_PSTBASOFF(R5)       ; SET END AS BASE FOR PST
 1467         ADDW    G^SWP$GW_EMPTPTE,R6     ; ASSUMES NO OVERFLOW POSSIBLE
 1468         ASHL    #7,R6,R6                ; CONVERT TO LONGWORD COUNT
 1469         MOVL    R6,PHD$L_WSLX(R5)       ; SET BASE OFFSET TO WSLX AREA
 1470         ADDL    G^SGN$GL_PTPAGCNT,R7    ; ADD PAGE TABLES TO COUNT
 1471         ADDL3   #1,R7,R0                ; ROUND TO LONGWORD SIZE
 1472 ;
 1473 ;       If we have a longword wslx array (mmg$gw_bigpfn equals 1)
 1474 ;       then we do not not bias the number of entries (in r0).  If
 1475 ;       we have a word wslx array, we divide by 2 to get the proper
 1476 ;       number of word length entries.  Note: phd$l_bak is store
 1477 ;       as an index, not a byte offset.
 1478 ;
 1479         PFN_DISP_IF_BIGPFN_THEN         END_BIGPFN_CODE=45$

                 ;This code executes if the PFN link arrays are longword arrays.
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 31
X-45U3 INITIALZE SHELL WITH SYSGEN PARAMETERS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELI

 1480          PFN_DISP_ELSE                   ELSE_CODE=45$,COMMON_CODE=46$

                ;This code executes if the PFN link arrays are word arrays.
 1481          DIVL    #2,R0                   ; set as wslx as word array
 1482          PFN_DISP_ENDIF                  COMMON_CODE=46$

                ;End of code that depends on size of PFN link arrays
 1483          ADDL    R0,R6                   ; ALLOCATE SPACE IN WHOLE LONGWORDS
 1484          MOVL    R6,PHD$L_BAK(R5)        ; SET BASE OF BACKING STORE VECTOR
 1485          ADDL    R7,R6                   ; ALLOCATE SPACE FOR BACKING STORE VECTOR
 1486          MULL    #4,R6                   ; CONVERT TO BYTE OFFSET
 1487          MOVL    R6,PHD$L_PTWSLELCK(R5)  ; SET BASE OF LOCKED COUNT VECTOR
 1488          MOVL    G^SGN$GL_PTPAGCNT,R7    ; GET COUNT OF PAGE TABLES
 1489          MOVAB   (R6)[R7],PHD$L_PTWSLEVAL(R5)    ; ALLOCATE AND SET BASE OF VAL CNT
 1490          MOVAB   @PHD$L_PTWSLEVAL(R5)[R5],R0     ; SET BASE FOR VALID COUNT VEC
 1491          MNEGL   #1,R2                           ; MINUS ONE FOR BACKGROUND
 1492          MOVAB   @PHD$L_PTWSLELCK(R5)[R5],R1     ; AND BASE FOR LOCKED COUNT
 1493 50$:     MOVB    R2,(R0)+                ; INIT BOTH COUNT VECTORS TO MINUS ONE
 1494          MOVB    R2,(R1)+                ;
 1495          SOBGTR  R7,50$                  ; FOR ALL PAGE TABLE SLOTS
 1496          ADDB    #KSTACK+1,-2(R1)        ; COUNT OF LOCK PAGES FOR SHELL
 1497          ADDB    #KSTACK+3,-2(R0)        ; COUNT OF VALID PAGES
 1498          MOVL    G^SGN$GL_PHDAPCNT,R6    ; GET ACTUAL HEADER SIZE
 1499          MOVZBL  G^SWP$GB_SHLP1PT,R0     ; NUMBER OF PERM PAGE TABLES
 1500          MOVW    R0,PHD$W_PTCNTLCK(R5)   ; COUNT OF PT CONTAINING LOCKED PAGES
 1501          MOVW    R0,PHD$W_PTCNTVAL(R5)   ; COUNT OF PT CONTAINING VALID PAGES
 1502          MOVW    R0,PHD$W_PTCNTACT(R5)   ; COUNT OF ACTIVE PAGE TABLES
 1503          MOVW    R0,PHD$W_PTCNTMAX(R5)   ; COUNT OF PT WITH NON-ZERO ENTRIES
 1504          ADDL    R0,R6                   ; MAKE ROOM IN WSLIST FOR HDR+PGTBLS
 1505          MOVL    PHD$L_WSDYN(R5),R7      ; GET INDEX TO CURRENT DYNAMIC ENTRIES
 1506          PUSHL   R7                      ; SAVE POINTER FOR LATER USE
 1507          ADDL    R6,PHD$L_WSLOCK(R5)     ; CORRECT LOCK WSL INDEX
 1508          ADDL    R6,PHD$L_WSDYN(R5)      ; AND DYNAMIC INDEX
 1509          ADDL    R6,PHD$L_WSNEXT(R5)     ; AND NEXT POINTER
 1510          ADDL3   R6,R7,R1                ; SET DESTINATION INDEX
 1511          MOVL    #SWP$C_NDYN,R0          ; SET COUNT OF DYNAMIC PAGES
 1512 60$:     MOVL    (R5)[R7],(R5)[R1]       ; MOVE A DYNAMIC WORKING SET PAGE
 1513          INCL    R1                      ; NEXT DESTINATION
 1514          INCL    R7                      ; NEXT WSL ENTRY
 1515          SOBGTR  R0,60$                  ; MOVE ALL DYNAMIC PAGES
 1516          MOVL    #<WSL$M_VALID-          ; WSL ENTRIES FOR HEADER ARE VALID
 1517                  !WSL$M_WSLOCK-          ; LOCKED,
 1518                  !WSL$C_PPGTBL-          ; PROCESS PAGE TABLE,
 1519                  !VA$M_SYSTEM>,R8        ; AND SYSTEM SPACE
 1520          MOVL    R5,R2                   ; SET PHD ADDRESS TO GET SVAPTE
 1521          JSB     G^MMG$SVAPTECHK         ; SVAPTE TO R3
 1522          MOVL    R8,R0                   ; MAKE WORKING COPY
 1523          MOVZWL  G^SWP$GW_WSLPTE,R1      ; COUNT OF WSLPTE
 1524          MOVL    (SP)+,R2                ; RESTORE WORKING SET INDEX TO LOCKED
 1525          BSBW    FILLPHD                 ; SET UP FOR VALID PHD PAGES
 1526          MOVZWL  G^SWP$GW_EMPTPTE,R1     ; GET COUNT OF EMPTY PTE AGAIN
 1527          MOVAL   (R3)[R1],R3             ; ADVANCE SPT ADDRESS
 1528          ASHL    #9,R1,R1                ; CONVERT TO BYTE COUNT
 1529          ADDL    R1,R0                   ; AND UPDATE CURRENT VA
 1530          MOVZWL  G^SWP$GW_BAKPTE,R1      ; GET COUNT OF REMAINING HEADER PAGES
 1531          BSBW    FILLPHD                 ; SET UP FOR VALID PHD PAGES
 1532          MOVL    G^SGN$GL_PTPAGCNT,R1    ; GET COUNT OF PAGE TABLES
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 32
X-45U3 INITIALZE SHELL WITH SYSGEN PARAMETERS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELI

 1533 65$:     MOVL    #PTE$C_ERKW,-4(R3)[R1]  ; SET A DEMAND ZERO PAGE FOR PAGE TABLES
 1534          SOBGTR  R1,65$                  ; INIT ENTIRE PAGE TABLE AREA
 1535          MOVZBL  G^SWP$GB_SHLP1PT,R1     ; GET COUNT OF P1 PAGE TABLES
 1536          MNEGL   R1,R6                   ; NEGATE COUNT
 1537          DECL    R6                      ; LESS ONE MORE FOR DEBUGGER PAGE
 1538          ADDL    G^SGN$GL_PTPAGCNT,R6    ; PAGE OFFSET TO PAGE TABLE
 1539          MOVAL   (R3)[R6],R3             ; COMPUTE SVAPTE
 1540          ASHL    #9,R6,R6                ; CONVERT TO BYTE OFFSET
 1541          ADDL    R6,R0                   ; ADD TO CURRENT BASE WSL
 1542          PUSHR   #^M<R1,R3>              ; SAVE KEY REGS
 1543          SUBL    S^#SWP$C_SHLP1PT,R1     ; LESS STANDARD P1 PAGE TABLES
 1544          BLEQ    80$                     ; BR IF NO EXTENSIONS
 1545 70$:     MOVL    -(R11),(R3)+            ; STORE A PTE FOR EXTENSION
 1546          SOBGTR  R1,70$                  ; DO ALL EXTENSION PAGE TABLES
 1547 80$:     MOVL    S^#SWP$C_SHLP1PT,R1     ; SET COUNT OF STANDARD
 1548 90$:     MOVL    (R9)+,(R3)+             ; STORE PTE FOR STANDARD
 1549          SOBGTR  R1,90$                  ; DO ALL STANDARD P1PT
 1550          POPR    #^M<R1,R3>              ; RESTORE KEY REGISTERS
 1551          BSBW    FILLPHD                 ; SET UP FOR VALID PHD PAGES
 1552          ROTL    #9,G^SGN$GL_PHDPAGCT,R0 ; OFFSET FOR P0BR
 1553          ADDL    R0,PHD$L_P0BR(R5)       ; FORM P0BR BASE VALUE
 1554          ROTL    #9,G^SGN$GL_PTPAGCNT,R1 ; OFFSET TO P1BR
 1555          ADDL    R0,R1                   ;
 1556          ADDL    R1,PHD$L_P1BR(R5)       ; FORM P1BR BASE VALUE
 1557          SUBL    R0,PHD$L_FREP1VA(R5)    ; COMPUTE CORRECT FREP1VA
 1558          SUBL    R0,R1                   ; EXTRACT PT SIZE TOTAL
 1559          ASHL    #-9,R0,R0               ; BACK TO PAGE COUNT
 1560          SUBL    R0,PHD$L_P1LR(R5)       ; NOW CORRECT P1 LENGTH REGISTER
 1561          ASHL    #-2,R1,R1               ; COUNT OF AVAILABLE PTES
 1562          SUBL    R0,R1                   ; DIMINISH BY PHD WINDOW SIZE
 1563          SUBL    #7,R1                   ; GUARANTEE 7 FREE PTES FOR TB PREFETCH
 1564                                          ;  HARDWARE
 1565          ADDL    R1,PHD$L_FREPTECNT(R5)  ; AND FORM COUNT OF FREE PTES
 1566
 1567 ; THE FOLLOWING SHUFFLE OF THE WORKING SET LIST REARRANGES ALL WORKING SET
 1568 ; LIST ENTRIES THAT ARE NOT KERNEL STACK PAGES. THE SWAPPER MAP IS BEING
 1569 ; REORDERED TO AGREE WITH THE WORKING SET LIST DEFINED IN THE SHELL.
 1570
 1571          .REPEAT SHUFFLE_COUNT           ; COUNT FROM WSL TEMPLATE
 1572          MOVL    (R9)+,(R11)+            ; REARRANGE SWAP PAGE LIST
 1573          .ENDR
 1574
 1575          CLRL    (R11)+                  ; SET STOPPER IN MAP
 1576          ADDL3   #512,PHD$L_FREP1VA(R5),-        ; SET ADDRESS OF WINDOW
 1577                  G^<SWP$AL_PTRPAG+-      ; INTO POINTER PAGE
 1578                  <CTL$GL_PHD-CTL$GL_VECTORS>>    ; THROUGH SWAPPER MAP
 1579          MOVL    R4,G^<SWP$AL_PTRPAG+-   ; SET PCB ADDRESS
 1580                  <CTL$GL_PCB-CTL$GL_VECTORS>>    ; THROUGH SWAPPER MAP
 1581          BBSS    #PCB$V_PHDRES,PCB$L_STS(R4),100$; MARK PHD RESIDENT
 1582 100$:                                    ;
 1583          MOVL    PHD$L_WSLIST(R5),R0     ; GET INDEX TO WS BASE
 1584          DECL    R0                      ;
 1585          ADDL3   R0,G^SGN$GL_MAXWSCNT,R1 ; Calculate maximum working set,
 1586          MOVL    R1,PHD$L_WSEXTENT(R5)   ; store as extent
 1587          MOVL    R1,PHD$L_WSAUTHEXT(R5)  ; store as authorized extent
 1588
 1589          ADDL3   R0,#<1@16>,R2           ; Limit working set quota to 64K pages
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 33
X-45U3 INITIALZE SHELL WITH SYSGEN PARAMETERS 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELI

 1590          CMPL    R1,R2                   ; ** This limit cannot be removed until
 1591          BLEQU   105$                    ; ** pagefile allocation routines can
 1592          MOVL    R2,R1                   ; ** handle arbitrary request sizes
 1593 105$:    MOVL    R1,PHD$L_WSQUOTA(R5)    ; store as working set quota
 1594          MOVL    R1,PHD$L_WSAUTH(R5)     ; store as authorized working set quota
 1595
 1596          MOVL    G^SGN$GL_MINWSCNT,PHD$L_WSFLUID(R5)     ; SET FLUID REQUIREMENT
 1597
 1598          MOVL    G^SGN$GL_DFWSCNT,R1                     ; GET DEFAULT WS SIZE
 1599          ADDL    R1,R0                                   ; CALC LAST
 1600          MOVL    R0,PHD$L_WSLAST(R5)                     ; SET LAST
 1601          MOVL    R0,PHD$L_DFWSCNT(R5)                    ; AND DEFAULT COUNT
 1602          MOVL    R1,PHD$L_WSSIZE(R5)                     ; SET WS SIZE
 1603          ADDL    R5,PHD$L_P0BR(R5)       ; BIAS P0 BASE REGISTER
 1604          ADDL    R5,PHD$L_P1BR(R5)       ; AND P1 BASE REGISTER ALSO
 1605 ;
 1606 ;        REMAP SYSTEM SERVICE VECTORS FOR THIS PROCESS
 1607 ;
 1608          MOVL    #P1SYSVECTORS,R2        ; PUT THEM HERE
 1609          JSB     G^MMG$SVAPTECHK         ; GET ADDRESS OF PTE FOR VECTORS
 1610          MOVL    G^SGN$A_COMPVALUES+MMG_GL_SPTBASE,R0    ; SYSTEM PAGE TABLE
 1611          MOVZBL  #SGN$C_SYSVECPGS,R1     ; NUMBER OF PAGES
 1612 110$:    BISL3   #PTE$M_WINDOW,(R0)+,(R3)+
 1613          SOBGTR  R1,110$
 1614 ;
 1615 ;        THE PAGE TABLE ARRAY FOR LOCKED WSLE'S MUST BE INCREMENTED ONE
 1616 ;        FOR EACH WINDOW PTE.  THIS BALANCES THE LOGIC IN $CREPAG/$DELPAG
 1617 ;        THAT PLACES THE PAGE TABLE PAGE IN THE LOCKED PORTION OF THE
 1618 ;        WORKING SET FOR USER PFNMAPPED PAGES.
 1619 ;
 1620 ;        THIS LOGIC ASSUMES THAT THE VECTOR PAGES ALL LIVE IN THE SAME
 1621 ;        PAGE TABLE PAGE.
 1622 ;
 1623          SUBL3   PHD$L_P0BR(R5),R3,R0    ;BYTE OFFSET OF PTE
 1624          ASHL    #-9,R0,R0               ;BYTE INDEX OF CONTAINING PAGE TABLE
 1625          ADDL    PHD$L_PTWSLELCK(R5),R0 ;COMPUTE THE ADDRESS OF THE LOCKED
 1626          ADDL    R5,R0                   ; WORKING SET LIST PAGE TABLE BYTE
 1627          ADDB    #SGN$C_SYSVECPGS,(R0)   ; ARRAY AND INCR BY NUMBER OF VEC PAGES
 1628
 1629 ; CLEAR THE PCB FIELDS THAT WERE USED TO CARRY PAGE FILE
 1630 ; INFORMATION TO THE NEW PROCESS. THESE FIELDS WILL BECOME THE COMMON EVENT
 1631 ; FLAG CLUSTER POINTERS.
 1632
 1633          .IF NDF PRMSW
 1634          ASSUME PCB$L_EFC3P EQ <PCB$L_EFC2P + 4>
 1635          ASSUME PCB$W_PGFLCHAR EQ PCB$L_EFC2P
 1636          ASSUME PCB$B_PGFLINDEX EQ <PCB$L_EFC2P + 2>
 1637          .ENDC
 1638
 1639          CLRQ    PCB$L_EFC2P(R4)         ; START WITH NO COMMON EF CLUSTERS
 1640
 1641          RSB                             ; RETURN
 1642
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 34
X-4503 FILLPHD - SETUP A VALID PHD PTE 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1

 1644          .SBTTL  FILLPHD - SETUP A VALID PHD PTE
 1645 ;
 1646 ;        R0 - WORKING SET LIST ENTRY, ADDRESS AND FLAGS (UPDATED)
 1647 ;        R1 - COUNT OF PAGES TO FILL (UPDATED)
 1648 ;        R2 - WORKING SET LIST INDEX (UPDATED)
 1649 ;        R3 - SVAPTE FOR PHD PAGE (UPDATED)
 1650 ;        R4 - PCB ADDRESS
 1651 ;        R5 - PHD ADDRESS
 1652 ;        R7 - SCRATCH
 1653 ;        R8 - SCRATCH
 1654 ;        R10 - PFN$C_ERKW!PFN$M_MODIFY!PFN$M_VALID
 1655 ;
 1656 FILLPHD:                                    ; SETUP VALID PHD PTES
 1657 10$:     MOVL    R0,(R5)[R2]                ; STORE WORKING SET LIST ENTRY
 1658          BICL3   R10,(R3)+,R7               ; GET PFN FROM SPT
 1659          PUSHL   G^SGN$A_COMPVALUES+PFN_AB_STATE          ; STACK ADDRESS BASE FOR PFN
 1660          MOVB    #<PFN$C_ACTIVE!PFN$M_MODIFY>,@(SP)+[R7] ; MARK PAGE ACTIVE
 1661          PUSHL   G^SGN$A_COMPVALUES+PFN_AB_TYPE           ; STACK ADDRESS BASE FOR PFN
 1662          MOVB    #PFN$C_PPGTBL,@(SP)+[R7]; SET PAGE TYPE TO PAGE TABLE
 1663          PUSHL   G^SGN$A_COMPVALUES+PFN_AL_BAK            ; STACK ADDRESS BASE FOR PFN
 1664          MOVZBL  PHD$B_PAGFIL(R5),R8        ; GET PAGE FILE NUMBER
 1665          ROTL    #PFN$V_PGFLX,R8,@(SP)+[R7]       ; SET BACKING STORE ADDRESS
 1666          PUSHL   G^SGN$A_COMPVALUES+PFN_AL_PTE            ; STACK ADDRESS OF PFN PTE B
 1667          MOVAL   -4(R3),@(SP)+[R7]          ; SET PTE BACK POINTER FOR PAGE
 1668 ;
 1669 ;  The following use of the PFN_REFERENCE macro must force absolute addressing
 1670 ;  because of the peculiar method in which this code executes. In addition,
 1671 ;  this code is not a part of the nonpaged executive and cannot have its opcode
 1672 ;  automatically fixed up by INIT.
 1673 ;
 1674          PUSHL   G^SGN$A_COMPVALUES+PFN_Ax_WSLX           ; STACK BASE ADDRESS OF WSLX
 1675                  PFN_REFERENCE   -
 1676          MOVW    <R2,@(SP)+[R7]>,-          ; SET WSLX FOR PAGE
 1677                  LONG_OPCODE=MOVZWL,-
 1678                  IMAGE=SHELL,-
 1679                  MODE=G^                    ; FORCE POSITION INDEPENDENCE
 1680          INCL    R2                         ; NEXT WORKING SET LIST ENTRY
 1681          MOVAL   512(R0),R0                 ; NEXT VA
 1682          SOBGTR  R1,10$                     ; FILL ALL REQUESTED PAGES
 1683          RSB                                ; AND RETURN
 1684
 1685 .IF      NDF     PRMSW
 1686 .IF      NDF     PROCESS_PAGE_DEFINITIONS
 1687          .ALIGN  PAGE                       ; PAGE ALIGN
 1688 .IFF
 1689          .=<.+511>&^XFE00
 1690 .ENDC
 1691 .ENDC
 1692
 1693 DEFINE_CONSTANT SWP$C_SHELLSIZ,<<.-SHELL>@-9>,SHELL ; SIZE OF SHELL PROCESS IN PAGES
 1694          .IF NDF PRMSW
 1695          .IF GT  SWP$C_SHELLSIZ-8
 1696          .ERROR  SWP$C_SHELLSIZ; Shell size changed, update swapper and FREELIM
 1697          .ENDC
 1698          .ENDC
 1699
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 35
X-45U3 FILLPHD - SETUP A VALID PHD PTE 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1

 1701
 1702 ;
 1703 ;       Process IFI/ISI Tables
 1704 ;
 1705
 1706 ; The rest of this module merely defines global symbols and offsets into
 1707 ; various P1 pages. No more storage is added to the SHELL module.
 1708
 1709         .PSECT  $ABS$,ABS
 1710
 1711         .=PIOTBL
 1712 TBL:
 1713         PIO2    A10                     ; LINK TO NEXT TABLE SEGMENT
 1714         .BLKL   1                       ; INITIALLY ZERO
 1715         .BLKL   IMP$C_NPIOFILES         ; IFAB TABLE SLOTS
 1716
 1717         PIO2    A20                     ; LINK TO NEXT TABLE SEGMENT
 1718         .BLKL   1                       ; INITIALLY ZERO
 1719         .BLKL   IMP$C_NPIOFILES         ; IRAB TABLE SLOTS
 1720
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 36
X-45U3 COMMAND LANGUAGE INTERPRETER DATA AREA 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELI

 1722          .SBTTL  COMMAND LANGUAGE INTERPRETER DATA AREA
 1723 ;------------------------------------------------------------------------------

 1724 ;                GENERIC COMMAND LANGUAGE INTERPRETER DATA
 1725 ;------------------------------------------------------------------------------

 1726
 1727          .=CLIDATAPAG                    ; SET BASE VALUE FOR REGION
 1728 DEFINE_SYMBOL CTL$AL_CLICALBK            ; CALL BACK VECTOR FOR CLI
 1729          .BLKL   1                       ;
 1730          .BLKL   1                       ; ALLOW FOR EXPANSION
 1731 DEFINE_SYMBOL CTL$AG_CLIMAGE
 1732          .BLKL   2                       ; VA RANGE INTO WHICH CLI IS MAPPED
 1733 DEFINE_SYMBOL CTL$AG_CLITABLE            ; VA RANGE INTO WHICH CLI TABLE IS
 1734          .BLKL   2                       ; MAPPED.
 1735
 1736 DEFINE_SYMBOL CTL$GL_UAF_FLAGS
 1737          .BLKL   1                       ; FLAGS FROM AUTHORIZATION RECORD
 1738
 1739 DEFINE_SYMBOL CTL$GT_CLINAME             ; CLI NAME (FILE NAME ONLY)
 1740          .BLKB   PQB$S_CLI_NAME          ; STORED AS COUNTED ASCII
 1741 DEFINE_SYMBOL CTL$GT_TABLENAME           ; CLI TABLE NAME (FULL FILE SPEC)
 1742          .BLKB   PQB$S_CLI_TABLE         ; STORED AS COUNTED ASCII
 1743 DEFINE_SYMBOL CTL$GT_SPAWNCLI           ; SPAWN CLI NAME (FILE NAME ONLY)
 1744          .BLKB   PQB$S_SPAWN_CLI         ; STORED AS COUNTED ASCII
 1745 DEFINE_SYMBOL CTL$GT_SPAWNTABLE         ; SPAWN CLI TABLE NAME (FULL FILE SPEC)
 1746          .BLKB   PQB$S_SPAWN_TABLE       ; STORED AS COUNTED ASCII
 1747
 1748 .IF NDF PRMSW
 1749 ASSUME <CTL$GT_TABLENAME -CTL$GT_CLINAME> EQ <PQB$T_CLI_TABLE  -PQB$T_CLI_NAME>
 1750 ASSUME <CTL$GT_SPAWNCLI  -CTL$GT_CLINAME> EQ <PQB$T_SPAWN_CLI  -PQB$T_CLI_NAME>
 1751 ASSUME <CTL$GT_SPAWNTABLE-CTL$GT_CLINAME> EQ <PQB$T_SPAWN_TABLE-PQB$T_CLI_NAME>
 1752 .ENDC
 1753
 1754 DEFINE_SYMBOL CTL$AG_CLIDATA                    ; START OF DATA AREA
 1755                                ; SIZE OF DATA AREA
 1756 DEFINE_CONSTANT CTL$C_CLIDATASZ,<CLIDATAEND-CTL$AG_CLIDATA>,SHELL
 1757
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 37
X-45U3 COMPATIBILITY MODE EMULATOR CONTEXT PAGE 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHE

 1759          .SBTTL  COMPATIBILITY MODE EMULATOR CONTEXT PAGE
 1760 ;-----------------------------------------------------------------------------

 1761 ;        COMPATIBILITY MODE EMULATOR CONTEXT PAGE
 1762 ;-----------------------------------------------------------------------------

 1763          .=CTL$AG_CMEDATA                ; POINT TO START OF AREA
 1764 DEFINE_SYMBOL CTL$AL_CMCNTX              ; COMPATIBILITY MODE REGISTER CONTEXT
 1765          .BLKL   10                      ; SAVED BY EXCEPTION
 1766
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 38
X-45U3 GLOBAL USER-MODE CONTEXT PAGE 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6

 1768          .SBTTL  GLOBAL USER-MODE CONTEXT PAGE
 1769 ;-----------------------------------------------------------------------------

 1770 ;        GLOBAL USER-MODE-ONLY CONTEXT PAGE
 1771 ;
 1772 ;        This page may ONLY be used by user-mode code, since the data
 1773 ;        has absolutely no protection from errant user-mode programs.
 1774 ;        Do NOT use this area for cells which cannot be wiped out by
 1775 ;        any user-mode program at any time during image execution,
 1776 ;        except when it only prevents that program from running.
 1777 ;-----------------------------------------------------------------------------

 1778          .=UWVECPAG                          ; POINT TO START OF AREA
 1779
 1780 DEFINE_SYMBOL CTL$GL_DCLPRSOWN                ; ADDRESS OF DCL WRK AREA CREATED
 1781          .BLKL   1                           ; BY DCL OR CLI$DCL_PARSE
 1782 DEFINE_SYMBOL CTL$GL_CLINTOWN                ; ADDRESS OF CONTEXT BLOCK USED BY
 1783          .BLKL   1                           ; CLI$INTERFACE PACKAGE IN DCL/MCR
 1784 DEFINE_SYMBOL CTL$GL_PSTFLAGS,PROCESS_SCHED ; Process start flags
 1785          .BLKL   1
 1786 DEFINE_SYMBOL CTL$GL_VVIEF_BASE,VOLATILE ; Base address of VVIEF
 1787          .BLKL   1
 1788 DEFINE_SYMBOL CTL$GL_VVIEF_END,VOLATILE ; End address of VVIEF
 1789          .BLKL   1
 1790 DEFINE_SYMBOL CTL$GL_VVIEF_ADDR,VOLATILE ; Transfer address of VVIEF
 1791          .BLKL   1
 1792 DEFINE_SYMBOL CTL$GL_VVIEF_FLAGS,VOLATILE ; Flags longword for VVIEF
 1793          .BLKL   1
 1794 DEFINE_SYMBOL CTL$GB_SOFT_AST_DISABLE,VOLATILE ; ADDRESS OF LONGWORD USED FOR
 1795          .BLKB   1                           ; FAST $SETAST FLAGS
 1796
 1797 ; THE REST OF THIS PAGE IS UNUSED
 1798
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 39
X-45U3 IMAGE ACTIVATOR CONTEXT PAGE 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

1800          .SBTTL  IMAGE ACTIVATOR CONTEXT PAGE
1801 ;*****************************************************************************

1802 ;       IMAGE ACTIVATOR CONTEXT PAGE
1803 ;
1804 ;       The following page contains image activator context that must
1805 ;       remain behind after an image is activated. The IAFLIST survives
1806 ;       across successive activations. The IAFEXE and IAFMERGE lists are
1807 ;       cleared when a new image activation is initiated. The FIXUP link
1808 ;       is cleared as soon as a given fixup pass completes.
1809 ;
1810 ;*****************************************************************************

1811          .=IMGACTCTX                    ; Point to start of area
1812
1813 DEFINE_SYMBOL CTL$GL_IAFLINK,IMAGE_ACTIVATOR ; Listhead of linked list of fixup vect
1814          .BLKL   1                      ;  (containing shareable image lists)
1815 DEFINE_SYMBOL CTL$GL_IAFLAST,IMAGE_ACTIVATOR ; Address of last fixup vector in list
1816          .BLKL   1                      ;  (used in normal and merged image activati
1817
1818 DEFINE_SYMBOL CTL$GL_FIXUPLNK,IMAGE_ACTIVATOR ; Listhead of linked list of fixup
1819          .BLKL   1                      ;  vectors used during fixup pass
1820
1821 DEFINE_SYMBOL CTL$GL_P1MERGE,IMAGE_ACTIVATOR ; Listhead of linked list of fixup vect
1822          .BLKL   1                      ;  for images merged into P1 space
1823
1824 ; The following data structure is an empty fixup vector that facilitates
1825 ; the fixup vector list manipulation. It is plugged into the fixup vector
1826 ; lists by PROCSTRT when a process is first created.
1827
1828 DEFINE_SYMBOL CTL$GL_IAFPERM,IMAGE_ACTIVATOR
1829 .BLKB   IAF$K_LENGTH
1830
1831 ; The following label locates the link field in the just allocated structure.
1832 ; While an image is active, it locates the fixup vector for an executable image.
1833 ; It is cleared as part of the image activation initialization code path.
1834
1835 DEFINE_CONSTANT CTL$GL_IAFEXE ,<CTL$GL_IAFPERM + IAF$L_IAFLINK>,-
1836                                 <IMAGE_ACTIVATOR,SHELL>
1837
1838 DEFINE_SYMBOL IAC$GL_IMAGCTX,IMAGE_ACTIVATOR
1839          .BLKL   1                      ; Context that exists for life of image
1840
1841 DEFINE_SYMBOL IAC$GL_PROCCTX,IMAGE_ACTIVATOR
1842          .BLKL   1                      ; Context that exists beyond image exit
1843
1844 DEFINE_SYMBOL IAC$AL_VECADDR,IMAGE_ACTIVATOR
1845          .BLKL   4                      ; Array of altered opcode addresses
1846
1847 DEFINE_SYMBOL IAC$AL_VECOPCOD,IMAGE_ACTIVATOR
1848          .BLKB   4                      ; Array of saved opcodes
1849
1850 DEFINE_SYMBOL IAC$AW_VECRESET,IMAGE_ACTIVATOR
1851          .BLKW   6                      ; Array of offsets used to reset vectors
1852
1853 DEFINE_SYMBOL IAC$AW_VECSET,IMAGE_ACTIVATOR
1854          .BLKW   6                      ; Array of offsets used to locate vectors
1855
1856 ; The following linked list contains image control blocks for all of the
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 40
X-45U3 IMAGE ACTIVATOR CONTEXT PAGE 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MAR;1 (6)

 1857 ; images currently mapped into a process' address space.
 1858
 1859 DEFINE_SYMBOL IAC$GL_IMAGE_LIST,IMAGE_ACTIVATOR
 1860         .BLKL   2                       ; Room for forward and backward links
 1861
 1862 ; The following linked list is used by the image activator to record work
 1863 ; in progress. It is empty while an image is executing.
 1864
 1865 DEFINE_SYMBOL IAC$GL_WORK_LIST,IMAGE_ACTIVATOR
 1866         .BLKL   2                       ; Room for forward and backward links
 1867
 1868 ; The following list is a potential source of unused image control blocks.
 1869 ; Although empty when a process is created, it grows to reflect the largest
 1870 ; number of images activated at the same time.
 1871
 1872 DEFINE_SYMBOL IAC$GL_ICBFL,IMAGE_ACTIVATOR
 1873         .BLKL   2                       ; Room for forward and backward links
 1874
 1875 ; The following two cells locate the ICB for the main image and the ICB for
 1876 ; the image most recently merged into the address space.
 1877
 1878 DEFINE_SYMBOL IAC$GL_MAIN_ICB,IMAGE_ACTIVATOR
 1879         .BLKL   1                       ; ICB of main image
 1880
 1881 DEFINE_SYMBOL IAC$GL_FIRST_ICB,IMAGE_ACTIVATOR
 1882         .BLKL   1                       ; ICB of image just merged
 1883
 1884 DEFINE_SYMBOL IAC$GL_STACK_SIZE,IMAGE_ACTIVATOR
 1885         .BLKL   1                       ; Amount by which to expand user stack
 1886
 1887 ; The following cell is a descriptor pointing to a list of identifiers, built
 1888 ; by the image acrivator.  These identifiers go away at image rundown.
 1889
 1890 DEFINE_SYMBOL IAC$GQ_IMAGE_IDS,IMAGE_ACTIVATOR
 1891         .BLKQ   1                       ; Your basic descriptor
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 41
X-45U3 Kernel mode, process OWN data pages 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MP

 1893          .SBTTL  Kernel mode, process OWN data pages
 1894 ;-----------------------------------------------------------------------------

 1895 ;       Kernel mode, process OWN data pages
 1896 ;
 1897 ;       The following pages are for use by VMS kernel mode
 1898 ;       facilities for process OWN storage. The decision to place a
 1899 ;       data cell here should be predicated on the same criteria
 1900 ;       used for the P1 pointer page. However, preference should be
 1901 ;       given to this location, since these pages are not permanently
 1902 ;       locked in the process working set.
 1903 ;
 1904 ;       Note that these pages are demand zero, implying that if non-zero
 1905 ;       initial contents are required, then the initialization must be
 1906 ;       done by code in EXE$PROCSTRT.
 1907 ;
 1908 ;-----------------------------------------------------------------------------

 1909
 1910          .=CTL$A_PRCPRM_KDATA            ; Point to start of area
 1911 KDATA_BEGIN=.
 1912
 1913 DEFINE_SYMBOL CTL$GL_LICENSE_CNT,VOLATILE ; LMF - count of licensed products
 1914          .BLKL   1                       ; currently active for this process
 1915
 1916 DEFINE_SYMBOL CTL$GQ_RMS_RUREC_LKSB,FILES_VOLUMES ; Recovery unit doorbell lock LKSB
 1917          .BLKQ   1
 1918 DEFINE_SYMBOL CTL$GL_RMS_RUJ_COUNT,FILES_VOLUMES ; Active recovery unit journal coun
 1919          .BLKL   1
 1920 DEFINE_SYMBOL CTL$GL_RMS_RUJLCK_FLINK,FILES_VOLUMES ; RU journal lock block (RUJLCK)
 1921          .BLKL   1                       ;  (note: initialized by RMS, not PROCSTRT)
 1922 DEFINE_SYMBOL CTL$GL_RMS_RUJLCK_BLINK,FILES_VOLUMES ; RU journal lock block (RUJLCK)
 1923          .BLKL   1                       ;  (note: initialized by RMS, not PROCSTRT)
 1924
 1925 DEFINE_SYMBOL CTL$GQ_JPICTX,MISC        ; JPI context segment queue header
 1926          .BLKQ   1
 1927 DEFINE_SYMBOL CTL$GL_JPICTXCNT,MISC     ; Count of allocated context segments
 1928          .BLKL   1
 1929 DEFINE_SYMBOL CTL$GL_UAICTX,SECURITY    ; Pointer to UAI context segment
 1930          .BLKL   1
 1931 DEFINE_SYMBOL CTL$GQ_MISC_P1_FLAGS,MISC ; Miscellaneous P1 flags. The values
 1932          .BLKL   2                       ; used are defined in $CTLP1FLAGSDEF in
 1933                                          ; SYSDEFAE.SDL.
 1934
 1935 DEFINE_SYMBOL CTL$GQ_RMCB_QUE,VOLATILE  ; Process list of resource managers
 1936          .BLKL   2
 1937 DEFINE_SYMBOL CTL$GL_CUR_XSCB,VOLATILE  ; Current default transaction pointer
 1938          .BLKL   1
 1939
 1940 DEFINE_SYMBOL CTL$GL_POSIX_1,MISC       ; POSIX cell 1
 1941          .BLKL   1
 1942 DEFINE_SYMBOL CTL$GL_POSIX_2,MISC       ; POSIX cell 2
 1943          .BLKL   1
 1944
 1945 KDATA_END=.
 1946 .IF NDF PRMSW
 1947 ASSUME  <KDATA_END - KDATA_BEGIN> LE <2*512> ;Kernel data area > 2 pages
 1948 .ENDC
```

```
SHELL 10-MAY-1989 17:02:05 VAX MACRO V5.0-8 Page 42
X-45U3 Kernel mode, process OWN data pages 6-OCT-1988 22:58:56 _$254$DUA55:[SYS.SRC]SHELL.MP

 1950
 1951
 1952 ;
 1953 ; These symbols (SYS$...) specify which cells in the P1 pointer page may
 1954 ; be used by any program, regardless of system version, without having
 1955 ; to link with SYS.STB (and thus, make it system dependant).
 1956 ;
 1957 ; These symbols will remain constant forever and will continue to be supported
 1958 ; from release to release.
 1959 ;
 1960
 1961
 1962 .IFT                                   ; FOR USRVECTOR MODULE ONLY
 1963
 1964
 1965         .MACRO  USRSYM  SYM,VALCHECK,CTLSYM
 1966         .IF     B,CTLSYM
 1967 SYS$'SYM == CTL$'SYM
 1968         .IF     NB,VALCHECK
 1969         .IF     NE,CTL$'SYM'-^X'VALCHECK'
 1970         .ERROR  ; *** CTL$'SYM' MUST EQ ^X'VALCHECK' TO REMAIN COMPATIBLE WITH PREVI
 1971         .ENDC
 1972         .ENDC
 1973         .IFF
 1974 SYS$'SYM == CTL$'CTLSYM
 1975         .IF     NB,VALCHECK
 1976         .IF     NE,CTL$'CTLSYM'-^X'VALCHECK'
 1977         .ERROR  ; *** CTL$'CTLSYM' MUST EQ ^X'VALCHECK' TO REMAIN COMPATIBLE WITH PR
 1978         .ENDC
 1979         .ENDC
 1980         .ENDC
 1981         .ENDM
 1982
 1983         .DISABLE TRACEBACK
 1984
 1985         USRSYM  GL_CMCNTX,7FFEFF88      ; Address of 2 pages of AME storage
 1986         USRSYM  GL_IAFLINK,7FFEFF8C,-   ; Points to IMGACT fixup listhead
 1987                         GL_IAFLNKPTR
 1988         USRSYM  GL_IMGLSTPTR,7FFEFE84   ; Points to ICM list (for debugger)
 1989
 1990
 1991 .ENDC                                  ; OF LIBSWITCH CONDITIONAL STARTED AT PSECT
 1992
 1993
 1994         .END
```

5 SYSPCNTRL.LIS

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 0
Table of contents

  (2)      150        DECLARATIONS
  (3)      236        EXE$SUSPND - SUSPEND SYSTEM SERVICE
  (4)      423        KERNEL AST THAT SUSPENDS PROCESS
  (5)      491        G^EXE$RESUME - RESUME SYSTEM SERVICE
  (6)      541        G^EXE$HIBER - HIBERNATE SYSTEM SERVICE
  (7)      598        G^EXE$WAKE - WAKE SYSTEM SERVICE
  (8)      681        G^EXE$NAMPID - CONVERT PROCESS NAME TO PID
  (9)      882        EXE$CHECK_PCB_PRIV      - Check ability of one process to affect anoth
  (10)     932        CWPS$CHECK_NODE - check epid for valid remote node
  (11)     997        EXE$xPID_TO_xxx - CONVERT PID TO OTHER PID OR PCB ADDRESS
  (12)    1160        G^EXE$SETPRN - SET PROCESS NAME
  (13)    1241        CWPS$PARSE_PRCNAM - CWPS PARSE PROCESS NAME
  (14)    1342        PARSE_NODE - PARSE NODE NAME
  (15)    1582        CWPS$PCNTRL     Cluster pcntrl dispatcher
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 1
X-14 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (1)

    1         .TITLE  SYSPCNTRL PROCESS CONTROL SERVICES
    2         .IDENT  'X-14'
    3
    4 ;
    5 ;*****************************************************************************
    6 ;*                                                                           *
    7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984, 1988 BY                            *
    8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
    9 ;*  ALL RIGHTS RESERVED.                                                     *
   10 ;*                                                                           *
   11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
   12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
   13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
   14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
   15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
   16 ;*  TRANSFERRED.                                                             *
   17 ;*                                                                           *
   18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
   19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
   20 ;*  CORPORATION.                                                             *
   21 ;*                                                                           *
   22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
   23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
   24 ;*                                                                           *
   25 ;*                                                                           *
   26 ;*****************************************************************************
   27
   28 ;++
   29 ; FACILITY: EXECUTIVE, PROCESS CONTROL SYSTEM SERVICES
   30 ;
   31 ; ABSTRACT:
   32 ;       THIS MODULE CONTAINS THE ROUTINES WHICH IMPLEMENT THE PROCESS
   33 ;       CONTROL SERVICES, SUSPEND, RESUME, HIBERNATE AND WAKE.
   34 ;
   35 ; AUTHOR:
   36 ;       R. HUSTVEDT
   37 ;
   38 ; MODIFIED BY:
   39 ;
   40 ;       X-14    PT00035         Pankaj Tandon           04-Apr-1989
   41 ;               Clear suspen bit necessarily on preemption exit
   42 ;               and eliminate state causing $suspnd to be short
   43 ;               circuited.
   44 ;
   45 ;       X-13    CWH5113         CW Hobbs                7-Dec-1988
   46 ;               Move the process name check, so that we look for a
   47 ;               process named TIGNES::LELAC on the local node before
   48 ;               we look for a process named LELAC on node TIGNES.
   49 ;               Remove group specification code from process naming.
   50 ;               Remove CWPS conditionals.
   51 ;               Streamline the path through EXE$NAMPID for the
   52 ;               current process, since 80 to 90% of the calls
   53 ;               to EXE$NAMPID default both parameters.
   54 ;               Check that EXE$NAMPID is called at IPL 0, and fix
   55 ;               some inconsistent IPL usage in CWPS code.
   56 ;               Use EXE$AR_UPCASE_DAT table to upcase nodenames.
   57 ;               Correct inconsistent case conventions.
```

```
;
;       X-12    CWH5112         CW Hobbs                25-Oct-1988
;               Remove special case treatment of node field of zero
;               being the local node.  Now that the services are
;               clusterwide, that could lead to a hit against the
;               wrong process.  The entire PID will have to be used.
;
;       X-11    SSA0001         Stan Amway              13-Sep-1988
;               Honor NOSUSPEND bit in $SUSPND.
;
;       X-10    CWH5110         CW Hobbs                29-Aug-1988
;               Add support for CWPS.
;
;       X-9     RNG5009         Rod Gamache             16-Jun-1987
;               Release SCHED spinlock before call to ALLOCIRP.
;
;       X-8     JWT0276         Jim Teague              15-Jan-1987
;               Fix what was a reference to pageable code after
;               returning from EXE$NAMPID.  Also, fix bug that
;               occurs when a process is resumed after a hard
;               suspend had overriden a soft suspend.
;
;       X-7     JWT0275         Jim Teague              12-Jan-1987
;               Fix bug in some error paths.
;
;       X-6     JWT0263         Jim Teague              11-Nov-1986
;               Implement SOFT SUSPEND: wait the process at SUPER
;               AST mode.  ASTs at EXEC mode or KERNEL mode thereby
;               become deliverable.  After delivery, the process
;               re-suspends itself at SUPER AST mode.
;
;       X-5     RNG0004         Rod Gamache             11-Nov-1986
;               Fix code broken by previous edit.
;
;       X-4     RNG0004         Rod Gamache             29-Oct-1986
;               Fix branch errors.
;
;       V04-001 SF04001         Stephen Fiorelli        24-Oct-1985
;               System_service macro used to declare entry point
;               and build system service descriptor block.
;               Added $SYSVECTORDEF.
;
;       V03-013 LJK0256         Lawrence J. Kenah       7-Dec-1983
;               Only allow ASTs if XQP thread is active. Clear SUSPEN bit
;               if pool allocation fails.
;
;       V03-012 CWH3012         CW Hobbs                27-Sep-1983
;               In G^EXE$IPID_TO_EPID treat a null IPID as a special case,
;               and return the null.
;
;       V03-011 LJK0250         Lawrence J. Kenah       31-Aug-1983
;               Set the SUSPEN bit before lowering IPL to zero to insure
;               that the PCB of the target process has not disappeared.
;
;               Make the SUSPND AST a regular kernel AST so that it properly
;               interlocks with the XQP. Include the interlocking code.
;
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 3
X-14 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (1)

  115 ;       V03-010 CWH1007         CW Hobbs                14-May-1983
  116 ;               Enable the storing of the actual cluster node info in the
  117 ;               high bits of the EPID.
  118 ;
  119 ;       V03-009 CWH1003         CW Hobbs                27-Apr-1983
  120 ;               Change pid conversion routines to do more checking
  121 ;               of pid against pids stored in PCB.  Make sure that
  122 ;               condition codes reflect final value in R0.
  123 ;
  124 ;       V03-008 LJK0197         Lawrence J. Kenah       25-Mar-1983
  125 ;               Insure that all success paths raise IPL to SYNCH.
  126 ;
  127 ;       V03-007 ACG0321         Andrew C. Goldstein,    24-Mar-1983  0:19
  128 ;               Allow non-privileged control over processes of equal UIC
  129 ;
  130 ;       V03-006 ROW0168         Ralph O. Weber           3-MAR-1983
  131 ;               Change W^ references to G^.
  132 ;
  133 ;       V03-005 CWH1002         CW Hobbs                19-Feb-1982
  134 ;               Modify G^EXE$NAMPID to use extended PIDs, add PID conversion
  135 ;               routines:
  136 ;                       G^EXE$IPID_TO_PCB               - internal pid to pcb addres
  137 ;                       G^EXE$EPID_TO_PCB               - extended pid to pcb addres
  138 ;                       G^EXE$IPID_TO_EPID      - internal pid to extended pid
  139 ;                       G^EXE$EPID_TO_IPID      - extended pid to internal pid
  140 ;
  141 ;       V03-004 LJK0188         Lawrence J. Kenah       22-Oct-1982
  142 ;               Do not allow processes that are being deleted to also
  143 ;               be suspended.
  144 ;
  145 ;       V03-003 KDM46395        Kathleen D. Morse       28-Jun-1982
  146 ;               Change word displacement to longword.
  147 ;
  148 ;---
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 4
X-14 DECLARATIONS 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (2)

  150         .SBTTL  DECLARATIONS
  151
  152 ;
  153 ; INCLUDE FILES:
  154 ;
  155
  156         $ACBDEF                         ; DEFINE AST CONTROL BLOCK
  157         $CLUBDEF                        ; CLUSTER BLOCK
  158         $CPUDEF                         ; PER-CPU DATA BLOCK OFFSETS
  159         $CSBDEF                         ; CLUSTER SYSTEM BLOCK
  160         $CWPSDEF                        ; CLUSTER-WIDE PROCESS SERVICE DEFINITIONS
  161         $DYNDEF                         ; DYN$C_ structure codes
  162         $IPLDEF                         ; IPL DEFINITIONS
  163         $LOGDEF                         ; Logical Name definitions
  164         $PCBDEF                         ; PCB OFFSET DEFINITIONS
  165         $PRDEF                          ; PROCESSOR REGISTER DEFS
  166         $PRIDEF                         ; PRIORITY INCREMENT DEFINITIONS
  167         $PRVDEF                         ; PRIVILEGE BIT DEFINITIONS
  168         $PSLDEF                         ; PSL definitions
  169         $RSNDEF                         ; RESOURCE NUMBER DEFINITIONS
  170         $SBDEF                          ; SYSTEM BLOCK DEFINITIONS
  171         $SFDEF                          ; Saved call frame definitions
  172         $SSDEF                          ; STATUS DEFINITIONS
  173         $STATEDEF                       ; SCHEDULER STATE DEFINITIONS
  174         $SYSVECTORDEF                   ; Define system service vector offsets
  175         $UICDEF                         ; UIC FORMAT DEFINITIONS for CWPS
  176 ;
  177 ; EQUATED SYMBOLS:
  178 ;
  179 ;
  180 PID=4                                  ; DISPLACEMENT TO PID ARGUMENT
  181 PIDADR=4                               ; DISPLACEMENT TO PID ARGUMENT
  182 PRCNAM=8                               ; DISPLACEMENT TO PROCESS NAME
  183 FLAGS=12                               ; Displacement to flags longword
  184
  185
  186 ;
  187 ; Make sure that we know that externals are intentional
  188 ;
  189         .DISABLE GLOBAL
  190 ;
  191 ; External data and addresses
  192 ;
  193         .EXTERNAL       BUG$_CWSERR, BUG$_SSRVEXCEPT, CLU$GL_CLUB, -
  194                         CLU$GL_CLUSVEC, CLU$GW_MAXINDEX, SCH$GL_MAXPIX, -
  195                         SCH$GL_PCBVEC, SCH$GL_PIXWIDTH, SCH$GQ_HIBWQ, -
  196                         SCH$GQ_SUSP, SCH$GW_LOCALNODE, EXE$AR_UPCASE_DAT, -
  197                         EXE$C_CMSTKSZ, SYS$SO_VECTOR_BASE
  198 ;
  199 ; External routines
  200 ;
  201         .EXTERNAL       EXE$ALLOCIRP, SCH$NEWLVL, SCH$QAST, SCH$RSE, SCH$RWAIT,-

  202                         SCH$WAIT, SCH$WAITK, SCH$WAKE, SMP$GET_CURPCB,-
  203                         CWPS$ALLOCATE_SRV, CWPS$SSND_PCNTRL_RQST
  204
  205 ;
  206 ; A macro to conditionally check current IPL.  If IPL is greater
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 5
X-14 DECLARATIONS 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (2)

  207 ; than the max permitted, a bugcheck is taken
  208 ;
  209
  210         .MACRO  CHECK_IPL       MAX_IPL_PERMITTED, ?L1
  211
  212                 .if NE, cwps$_scared
  213                 mfpr    s^#pr$_ipl, -(SP)               ; Push the current IPL
  214                 cmpl    #<MAX_IPL_PERMITTED>, (SP)+     ; Is it what we want?
  215                 bgequ   L1                              ; OK, we can continue
  216                 bug_check CWSERR,FATAL                  ; Too high, make noise
  217         L1:     .endc
  218
  219         .ENDM
  220
  221         .MACRO  CHECK_IPL_EQL   IPL_REQUIRED, ?L1
  222
  223                 .if NE, cwps$_scared
  224                 mfpr    s^#pr$_ipl, -(SP)               ; Push the current IPL
  225                 cmpl    #<IPL_REQUIRED>, (SP)+          ; Is it what we want?
  226                 beql    L1                              ; OK, we can continue
  227                 bug_check CWSERR,FATAL                  ; Too high, make noise
  228         L1:     .endc
  229
  230         .ENDM
  231
  232
  233         DECLARE_PSECT   EXEC$NONPAGED_CODE ; NONPAGED EXEC
  234
```

```
236          .SBTTL  EXE$SUSPND - SUSPEND SYSTEM SERVICE
237 ;++
238 ;       G^EXE$SUSPND - SUSPEND SYSTEM SERVICE
239 ;
240 ; FUNCTIONAL DESCRIPTION:
241 ;       G^EXE$SUSPND IMPLEMENTS THE SUSPEND PROCESS SYSTEM SERVICE.
242 ;       THIS SERVICE CAUSES THE SPECIFIED PROCESS TO BE SUSPENDED
243 ;       BY INITIATING A KERNEL MODE AST IF NOT THE CURRENT PROCESS.
244 ;       A SUSPENDED PROCESS CANNOT RECEIVE ASTS AND WILL ONLY BE
245 ;       RESUMED AS A RESULT OF THE RESUME SYSTEM SERVICE OR A
246 ;       DELETE PROCESS REQUEST.
247 ;
248 ;
249 ; CALLING SEQUENCE:
250 ;       CALLG   ARGLIST,G^EXE$SUSPND
251 ;
252 ;
253 ; INPUT PARAMETERS:
254 ;       04(AP) - PROCESS IDENTIFICATION POINTER (PID)
255 ;       08(AP) - PROCESS NAME DESCRIPTOR POINTER
256 ;       12(AP) - FLAGS input.  Bit # 0 being set
257 ;                       indicates that this should
258 ;                       be a HARD suspend against
259 ;                       delivery of all ASTs.
260 ;
261 ;       R4 - PCB ADDRESS OF CURRENT PROCESS
262 ;
263 ; IMPLICIT INPUTS:
264 ;       PCB OF CURRENT PROCESS
265 ;       PCB OF TARGET PROCESS
266 ;
267 ;
268 ; OUTPUT PARAMETERS:
269 ;       R0 - COMPLETION STATUS
270 ;       @PID(AP) - PROCESS IDENTIFICATION OF TARGET PROCESS
271 ;
272 ; COMPLETION CODES:
273 ;       SS$_NORMAL  -  NORMAL SUCCESSFUL COMPLETION
274 ;       SS$_NOPRIV  -  INSUFFICIENT PRIVILEGE FOR REQUESTED OPERATION
275 ;       SS$_NONEXPR -  NON-EXISTENT PROCESS
276 ;       SS$_ACCVIO  -  ACCESS VIOLATION ON WRITE DESTINATION
277 ;       SS$_INSFMEM -  INSUFFICIENT DYNAMIC MEMORY FOR REQUEST
278 ;                       ( ONLY RETURNED IF NO RESOURCE WAIT ENABLE )
279 ;
280 ; SIDE EFFECTS:
281 ;       NONE
282 ;
283 ;--
284 CHECK_PRIV:
285          MOVPSL  R2                      ; Get mode of caller
286          EXTZV   #PSL$V_PRVMOD,-
287                  #PSL$S_PRVMOD,R2,R3     ; R3 = mode of caller
288          CMPB    R3,#PSL$C_EXEC          ; Was caller at least in EXEC mode?
289          BLEQU   SET_KERNEL              ; Take branch if KERNEL mode
290
291          MOVZWL  #SS$_NOPRIV,R0          ; If none of above, NOPRIV
292          BRB     EXIT_HELPER             ; Exit with R0
```

```
293
294 CANT_READ:
295         MOVZWL  #SS$_ACCVIO,R0          ; Can't read FLAGS arg
296
297 EXIT_HELPER:
298         BRW     EXIT                    ; Exit with R0
299
300 SET_KERNEL:
301         ASSUME  PSL$C_KERNEL EQ 0
302         CLRB    R5      ; access mode of this request is kernel
303         BRB     COMMON
304
305 NOSUSPEND:
306         MOVZWL  #SS$_NOSUSPEND,R0
307         BRW     EXIT_R0
308 ;
309 ; Process is being deleted
310 ;
311 DELPEN: MOVZWL  #SS$_NONEXPR,R0         ; RETURN NO SUCH PROCESS IF DELPEN
312         BRW     EXIT_R0                 ; EXIT AND UNLOCK SCHED
313
314         SYSTEM_SERVICE  SUSPND,-
315                         <R2,R3,R4,R5,R6>,-
316                         MODE=KERNEL,-
317                         NARG=2
318
319         MOVZBL  #PSL$C_SUPER,R5         ; R5 = access mode of this request
320                                         ;  (assume SUPER mode for now)
321         MOVL    SF$L_SAVE_FP(FP),R3     ; Get previous call frame
322         MOVL    SF$L_SAVE_PC(R3),R6     ; Find out from whence we came
323
324         CMPB    (AP),#2                 ; More than 2 args?
325         BLEQU   COMMON
326
327         IFNORD  #4,FLAGS(AP),CANT_READ  ; Can we read 3rd arg?
328         BLBS    FLAGS(AP),CHECK_PRIV    ; HARD bit set?
329
330         ; The SCHED DATABASE is locked by EXE$NAMPID
331
332 COMMON:
333         BSBW    EXE$NAMPID              ; Translate and verify args
334
335         ASSUME   DYN$C_CWPS_SUSPND  EQ  CWPSSRV$K_SUSPND
336         MOVZBL  #DYN$C_CWPS_SUSPND, R3  ; Load subtype for CWPS
337
338         BLBC    R0,EXIT_HELPER          ; Exit with error in R0
339
340         PUSHL   R1                                      ; Save PID
341         BBS     #PCB$V_DELPEN,PCB$L_STS(R4),-           ; Exit if being deleted
342                 DELPEN
343         BBS     #PCB$V_NOSUSPEND,PCB$L_STS(R4),-        ; BR if suspend not allowed
344                 NOSUSPEND
345         BBCS    #PCB$V_SUSPEN,PCB$L_STS(R4),100$        ; Already suspended?
346         BBC     #PCB$V_SOFTSUSP,PCB$L_STS(R4),150$      ; Soft suspend?
347
348 100$:   TSTB    R5                                      ; Hard suspend?
349         BEQL    160$                                    ; Take branch if so
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 8
X-14 EXE$SUSPND - SUSPEND SYSTEM SERVICE 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (3)

  350
  351 110$:   MOVAB   W^SUSPEND_SOFT_RETURN,R2              ; Soft suspend request
  352         CMPL    R2,R6                                 ; Did we get here from the
  353                                                       ;    SUPER AST?
  354         BEQL    130$                                  ; If so, take branch
  355         ASSUME  PCB$V_SOFTSUSP GE 24
  356         BISB    #<1@<PCB$V_SOFTSUSP-24>>,-
  357                   PCB$L_STS+3(R4)             ; If not, set SOFT SUSPEND bit
  358         UNLOCK  LOCKNAME=SCHED,-                ; unlock sched spinlock
  359                 PRESERVE=NO                     ; don't preserve R0
  360
  361         JSB     G^EXE$ALLOCIRP          ; allocate i/o packet for ast
  362         BLBC    R0,140$
  363
  364 120$:   MOVL    R2,R5                   ; setup pointer to ast control blk
  365
  366         MOVAL   W^SUSPEND_SOFT,ACB$L_AST(R5)    ; set for super ast on process
  367         MOVB    #PSL$C_SUPER,ACB$B_RMOD(R5)     ; set access mode for ast
  368         MOVL    (SP)+,ACB$L_PID(R5)             ; set pid for ast
  369         CLRL    R2                              ; set null priority increment
  370         BRB     170$
  371
  372 130$:   BBS     #PCB$V_PREEMPTED,PCB$L_STS(R4),131$
  373
  374         ASSUME  PCB$V_SOFTSUSP GE 24
  375         BISB    #<1@<PCB$V_SOFTSUSP-24>>,-
  376                   PCB$L_STS+3(R4)             ; If not, set SOFT SUSPEND bit
  377                                               ; thus enable suspension
  378
  379 131$:   BBC     #PCB$V_SOFTSUSP,PCB$L_STS(R4),190$ ; If soft suspend
  380
  381         BBSC    #PCB$V_RESPEN,PCB$L_STS(R4),180$        ;  and no RESUMEs pending
  382         MOVAQ   G^SCH$GQ_SUSP,R2                ;  then set up to wait
  383         MOVL    FP,SP                           ;  in mode of caller.
  384         MOVQ    8(SP),AP                        ; Clean stack first
  385         ADDL    S^#EXE$C_CMSTKSZ,SP             ; Back up PC (now points to
  386         SUBL    #4,(SP)                         ;    CHMK code)
  387         JMP     G^SCH$WAITK                     ; Wait...
  388
  389 140$:   BRW     EXIT_NO_POOL
  390
  391 150$:   BRW     EXITU
  392
  393 ;  HARD SUSPND code (old style $SUSPND)
  394
  395 160$:   UNLOCK  LOCKNAME=SCHED,-                ; unlock sched spinlock
  396                 PRESERVE=NO                     ; don't preserve R0
  397
  398         JSB     G^EXE$ALLOCIRP          ; ALLOCATE I/O PACKET FOR AST
  399         BLBC    R0,140$                 ; IF LBC THEN NO PACKET ALLOCATED
  400
  401         MOVL    R2,R5                   ; SETUP POINTER TO AST CONTROL BLK
  402
  403         MOVAL   B^SUSPND,ACB$L_AST(R5)  ; QUEUE KERNEL AST TO SUSPND:
  404         CLRB    ACB$B_RMOD(R5)          ; KERNEL MODE
  405         MOVL    (SP)+,ACB$L_PID(R5)     ; SET PID FOR AST
  406         CLRL    R2                      ; SET NULL PRIORITY INCREMENT
```

```
  407
  408 170$:   JSB     G^SCH$QAST                  ; QUEUE AST
  409         BRW     EXITN                       ; Exit with normal status
  410
  411 180$:   BICL    #<<1@PCB$V_SOFTSUSP>!<1@PCB$V_SUSPEN>>,PCB$L_STS(R4)
  412         BRW     EXITU
  413
  414 190$:   BBCC    #PCB$V_SUSPEN,-             ; Clear suspen bit anyway
  415                 PCB$L_STS(R4),195$
  416 195$:   BBCC    #PCB$V_PREEMPTED,-          ; Clear preempted and dismiss
  417                 PCB$L_STS(R4),200$          ; ast
  418
  419
  420 200$:   BRW     EXITU                       ; Exit with normal status
  421
```

```
423          .SUBTITLE       KERNEL AST THAT SUSPENDS PROCESS
424 ;+
425 ;
426 ; KERNEL AST ROUTINE TO SUSPEND PROCESS
427 ;
428 ; CALLING SEQUENCE:
429 ;       (SAME EFFECT AS) DCLAST ASTADR=DELETE MODE=KERNEL
430 ;
431 ; INPUT PARAMETERS:
432 ;       NONE
433 ;
434 ; OUTPUT PARAMETERS:
435 ;       NONE
436 ;
437 ; IMPLICIT INPUTS:
438 ;       PCB OF CURRENT PROCESS LOCATED VIA CPU$L_CURPCB IN PER-CPU DATA
439 ;
440 ; IMPLICIT OUTPUTS:
441 ;       PCB$V_SUSPEN - CLEARED
442 ;                                   > WHEN PROCESS IS RESUMED
443 ;       PCB$V_RESPEN - CLEARED
444 ;-
445
446          .ENABLE          LOCAL_BLOCK
447
448 SUSPND:                                     ; SUSPEND KERNEL AST ROUTINE
449          .WORD    ^M<R2,R3,R4,R5,R6>        ; SAVE SOME REGISTERS
450          JSB      G^SMP$GET_CURPCB          ; Get our PCB address
451
452 10$:     MOVPSL   -(SP)                     ; SAVE PSL ON STACK
453          LOCK     LOCKNAME=SCHED,-          ; LOCK SCHED DATABASE
454                   PRESERVE=NO               ; DON'T PRESERVE R0
455
456          BBSC     #PCB$V_SOFTSUSP,PCB$L_STS(R4),300$
457
458 310$:    BBCC     #PCB$V_RESPEN,PCB$L_STS(R4),30$ ; BR IF NO PENDING RESUME
459          UNLOCK   LOCKNAME=SCHED,-          ; UNLOCK SCHED DATBASE
460                   NEWIPL=#0,-               ; DROP IPL
461                   PRESERVE=NO               ; DON'T PRESERVE R0
462 EXIT_NO_POOL:
463          BBCC     #PCB$V_SUSPEN,PCB$L_STS(R4),20$ ; CLEAR SUSPEND PENDING
464 20$:     RET                                ; AND EXIT
465
466 30$:     TSTB     PCB$B_DPC(R4)             ; TEST FOR OUTSTANDING XQP ACTIVITY
467          BEQL     40$                       ; BRANCH IF NONE (ALLOW SUSPENSION)
468          BICB2    #1,PCB$B_ASTACT(R4)       ; CLEAR KERNEL AST ACTIVE
469          JSB      G^SCH$NEWLVL              ; COMPUTE NEW AST LEVEL
470          MOVL     #RSN$_ASTWAIT,R0          ; NOTE AST RESOURCE
471          JSB      G^SCH$RWAIT               ; WAIT FOR AST
472          BRB      10$                       ; MAKE THE TEST AGAIN
473
474 40$:     MOVAL    G^SCH$GQ_SUSP,R2          ; GET QUEUE HEADER ADDRESS
475          JSB      G^SCH$WAITK               ; WAIT WITH CLEAN STACK
476          BRB      10$                       ; AND CLEAR RESUME PENDING FLAG
477
478 300$:    BBSS     #PCB$V_PREEMPTED,-        ; Set preempted bit as state to
479                   PCB$L_STS(R4),320$        ;  be used in active SUPER thread
```

```
  480 320$:   BRB     310$                    ;  when it is resumed.
  481
  482 SUSPEND_SOFT:
  483         .WORD   ^M<R2,R3,R4,R5,R6>      ; SUPER AST routine
  484         $SUSPND_S                       ; Get back into $SUSPND in kernel
  485 SUSPEND_SOFT_RETURN:                    ;   mode, where we can deal with
  486         RET                             ;   all this
  487
  488         .DISABLE        LOCAL_BLOCK
  489
```

```
491          .SBTTL  G^EXE$RESUME - RESUME SYSTEM SERVICE
492 ;++
493 ;       G^EXE$RESUME - RESUME SYSTEM SERVICE
494 ;
495 ; FUNCTIONAL DESCRIPTION:
496 ;       G^EXE$RESUME IMPLEMENTS THE RESUME SYSTEM SERVICE WHICH RESTARTS
497 ;       A SUSPENDED PROCESS.
498 ;
499 ; INPUT PARAMETERS:
500 ;       04(AP) - PROCESS IDENTIFICATION POINTER (PID)
501 ;       08(AP) - PROCESS NAME DESCRIPTOR POINTER
502 ;       R4 - PCB ADDRESS OF CURRENT PROCESS
503 ;
504 ; IMPLICIT INPUTS:
505 ;       PCB OF CURRENT PROCESS
506 ;       PCB OF TARGET PROCESS
507 ;       PROCESS HEADER OF CURRENT PROCESS
508 ;
509 ; OUTPUT PARAMETERS:
510 ;       R0 - COMPLETION STATUS
511 ;       @PID - PROCESS IDENTIFICATION OF TARGET PROCESS
512 ;
513 ; IMPLICIT OUTPUTS:
514 ;       NONE
515 ;
516 ; COMPLETION CODES:
517 ;       SS$_NORMAL  -  NORMAL SUCCESSFUL COMPLETION
518 ;       SS$_NOPRIV  -  INSUFFICIENT PRIVILEGE FOR REQUESTED OPERATION
519 ;       SS$_NONEXPR -  NON-EXISTENT PROCESS
520 ;       SS$_ACCVIO  -  ACCESS VIOLATION ON WRITE DESTINATION
521 ;
522 ; SIDE EFFECTS:
523 ;       NONE
524 ;--
525
526          SYSTEM_SERVICE  RESUME,-
527                          <R2,R3,R4,R5>,-
528                          MODE=KERNEL,-
529                          NARG=2
530
531          BSBW    EXE$NAMPID              ; CONVERT AND VALIDATE
532          ASSUME  DYN$C_CWPS_RESUME   EQ  CWPSSRV$K_RESUME
533          MOVZBL  #DYN$C_CWPS_RESUME, R3  ; LOAD SUBTYPE FOR CWPS
534          BLBC    R0,EXIT                 ; EXIT IF ERROR OCCURRED
535          MOVZBL  #PRI$_RESAVL,R2         ; SET PRIORITY INCREMENT CLASS
536          BBSS    #PCB$V_RESPEN,PCB$L_STS(R4),10$ ; SET RESUME PENDING
537 10$:    RPTEVT  RESUME                  ; REPORT RESUME EVENT
538          BRB     EXITU                   ; AND TAKE NORMAL EXIT AND
539                                          ; UNLOCK SCHED DATABASE
```

```
541          .SBTTL  G^EXE$HIBER - HIBERNATE SYSTEM SERVICE
542 ;++
543 ;        G^EXE$HIBER - HIBERNATE SYSTEM SERVICE
544 ;
545 ; FUNCTIONAL DESCRIPTION:
546 ;        G^EXE$HIBER IMPLEMENTS THE HIBERNATE SYSTEM SERVICE WHICH
547 ;        PLACES THE PROCESS IN A WAIT STATE, HIB , UNTIL IT
548 ;        IS RE-AWAKENED BY A WAKE SYSTEM SERVICE.  ASTS MAY BE DELIVERED
549 ;        WHILE THE PROCESS IS IN A HIBERNATE STATE.
550 ;
551 ;
552 ;
553 ; CALLING SEQUENCE:
554 ;        CALLG   ARGLIST,G^EXE$HIBER
555 ;
556 ;
557 ; INPUT PARAMETERS:
558 ;        R4 - PCB ADDRESS OF CURRENT PROCESS
559 ;
560 ; IMPLICIT INPUTS:
561 ;        PROCESS CONTROL BLOCK(PCB) OF THE PROCESS ISSUING THE HIBERNATE
562 ;        SYSTEM SERVICE.
563 ;
564 ;
565 ; OUTPUT PARAMETERS:
566 ;        R0 - COMPLETION STATUS CODE
567 ;
568 ; IMPLICIT OUTPUTS:
569 ;        NONE
570 ;
571 ; COMPLETION CODES:
572 ;        SS$_NORMAL - NORMAL SUCCESSFUL COMPLETION
573 ;
574 ; SIDE EFFECTS:
575 ;        THE PROCESS WILL BE PLACED IN A WAIT STATE UNTIL EITHER
576 ;        AN AST IS DELIVERED OR A WAKE REQUEST IS MADE.
577 ;
578 ;--
579
580
581          SYSTEM_SERVICE  HIBER,-
582                          <R2,R3,R4,R5>,-
583                          MODE=KERNEL,-
584                          NARG=0
585
586          LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
587                  PRESERVE=NO             ; DON'T PRESERVE R0
588          BBCCI   #PCB$V_WAKEPEN,PCB$L_STS(R4),10$ ; CHECK FOR PENDING WAKE
589          UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATABASE
590                  NEWIPL=#IPL$_ASTDEL,-   ; DROP IPL
591                  PRESERVE=NO             ; DON'T PRESERVE R0
592          BRB     EXITN                   ; AND RETURN TO CALLER
593
594 10$:                                    ; MUST HIBERNATE
595          MOVAL   G^SCH$GQ_HIBWQ,R2       ; SET ADDRESS OF WAIT QUEUE HDR
596          JMP     G^SCH$WAIT              ; AND WAIT
```

```
598          .SBTTL  G^EXE$WAKE - WAKE SYSTEM SERVICE
599 ;++
600 ;        G^EXE$WAKE - WAKE SYSTEM SERVICE
601 ;
602 ; FUNCTIONAL DESCRIPTION:
603 ;        THE WAKE SYSTEM SERVICE CAUSES A PROCESS IN A HIBERNATE STATE
604 ;        TO BE CHANGED TO AN EXECUTABLE STATE AND RE-EXECUTED.
605 ;        IF THE TARGET OF A WAKE SERVICE IS NOT CURRENTLY HIBERNATING,
606 ;        THEN A BIT IS POSTED WHICH WILL CAUSE A SUBSEQUENT HIBERNATE
607 ;        CALL BY THAT PROCESS TO RETURN IMMEDIATELY.
608 ;
609 ; CALLING SEQUENCE:
610 ;        CALLG   ARGLIST,G^EXE$WAKE
611 ;
612 ; INPUT PARAMETERS:
613 ;        04(AP) = PROCESS IDENTIFICATION (PID) OF PROCESS TO WAKE
614 ;        08(AP) = ADDRESS OF PROCESS NAME DESCRIPTOR
615 ;        R4 - PCB ADDRESS
616 ;
617 ; IMPLICIT INPUTS:
618 ;        PCB OF CURRENT PROCESS
619 ;        ALL PCBS LOCATED BY THE VECTOR @SCH$GL_PCBVEC
620 ;
621 ; OUTPUT PARAMETERS:
622 ;        R0 - COMPLETION STATUS CODE
623 ;        @PID(AP) - PROCESS IDENTIFICATION (PID) OF PROCESS AWAKENED
624 ;
625 ; IMPLICIT OUTPUTS:
626 ;        PCB$V_WAKEPEN BIT IN PCB$L_STS OF TARGET PROCESS WILL BE
627 ;        SET IF PROCESS IS NOT HIBERNATING.
628 ;
629 ; COMPLETION CODES:
630 ;        SS$_NORMAL - NORMAL SUCCESSFUL COMPLETION
631 ;        SS$_NONEXPR - NON-EXISTENT PROCESS
632 ;        SS$_NOPRIV - NO PRIVILEGE FOR ATTEMPTED OPERATION
633 ;        SS$_ACCVIO - ACCESS VIOLATION ON WRITE DESTINATION
634 ;
635 ; SIDE EFFECTS:
636 ;        THE TARGET PROCESS WILL BE CHANGED TO AN EXECUTABLE STATE,
637 ;        COM OR COMO, IF IT IS IN A HIBERNATE STATE AND
638 ;        RESCHEDULING WILL BE INITIATED IF NECESSARY.
639 ;
640 ;--
641          SYSTEM_SERVICE  WAKE,-
642                          <R2,R3,R4,R5>,-
643                          MODE=KERNEL,-
644                          NARG=2
645
646          BSBB    EXE$NAMPID              ; CONVERT NAME TO PID
647 ;
648 ;        R0 - SUCCESS INDICATOR
649 ;        R1 - PID CORRESPONDING TO NAME STRING
650 ;        R4 - PCB ADDRESS IF NAME WAS FOUND
651 ;
652          ASSUME  DYN$C_CWPS_WAKE   EQ  CWPSSRV$K_WAKE
653          MOVZBL  #DYN$C_CWPS_WAKE, R3    ; LOAD SUBTYPE FOR CWPS
654
```

```
655          BLBC    R0,EXIT                 ; CONTINUE IF PROCESS LOCATED
656          JSB     G^SCH$WAKE              ; WAKE PROCESS BY PID
657 EXITU:                                   ; EXIT AND UNLOCK SCHED DATABASE
658          UNLOCK  LOCKNAME=SCHED,-         ; UNLOCK SCHED DATABASE -
659                  NEWIPL=#IPL$_ASTDEL,-    ; DROP IPL
660                  PRESERVE=NO              ; DON'T PRESERVE R0
661                                           ;  SCHED WAS LOCKED BY EXE$NAMPID
662 EXITN:                                   ; EXIT HIBERNATE SERVICE
663          MOVZWL  #SS$_NORMAL,R0           ; SET NORMAL COMPLETION
664 EXIT:                                    ; RETURN WITH R0 SET
665          CMPW    #SS$_REMOTE_PROC,R0      ; IS IT A CALL TO A REMOTE PROCESS?
666          BNEQ    10$                      ; NOT REMOTE, CONTINUE WITH ERROR
667          BRW     CWPS$PCNTRL              ; BRANCH TO THE CLUSTER CODE
668 10$:
669
670          SETIPL  #0                       ; ENABLE
671          RET                              ; AND RETURN TO CALLER
672
673 EXIT_R0:
674          UNLOCK  LOCKNAME=SCHED,-         ; UNLOCK SCHED DATABASE -
675                  NEWIPL=#0,-              ; DROP IPL
676                  PRESERVE=YES             ; PRESERVE R0
677          RET
678
679
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 16
X-14 G^EXE$NAMPID - CONVERT PROCESS NAME TO P 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (

        .SBTTL  G^EXE$NAMPID - CONVERT PROCESS NAME TO PID
;++
;       G^EXE$NAMPID  -  CONVERT PROCESS NAME TO PID
;
; FUNCTIONAL DESCRIPTION:
;       G^EXE$NAMPID OBTAINS THE PROPER PID AND PCB ADDRESS FOR A
;       STANDARD PROCESS CONTROL SERVICE ARGUMENT LIST CONSISTING
;       OF A PID/PROCESS-NAME PAIR.  THE ABSENCE OF BOTH SELECTS THE
;       CURRENT PROCESS.  AFTER ANY NECESSARY NAME TRANSLATION AND
;       PID VALIDATION, GROUP AND WORLD PROCESS CONTROL PRIVILEGES
;       ARE CHECKED.
;
;
; CALLING SEQUENCE:
;       JSB/BSB G^EXE$NAMPID
;
; INPUT PARAMETERS:
;       PID(AP) - ADDRESS OF PID SOURCE/DESTINATION (EXTENDED PID)
;       PRCNAM(AP) - POINTER TO PROCESS DESCRIPTOR TO CONVERT TO PID
;       R4 - PCB ADDRESS
;
;
;     ************************************************************************
;     *   N.B.                                                               *
;     *   Module <SYSLOA>CWPS_SERVICE_RECV.MAR calls EXE$NAMPID with         *
;     *   R4 pointing to a false PCB.  This trojan PCB is set up to          *
;     *   look like a valid PCB, but with the rights of a process on         *
;     *   a remote node.                                                     *
;     *                                                                      *
;     *   Changes to the calling sequence of EXE$NAMPID, or accesses         *
;     *   to additional fields of the PCB (or entities hanging from          *
;     *   the PCB) will have to be reflected in the code which sets          *
;     *   up this trojan PCB.                                                *
;     *                                                                      *
;     ************************************************************************
;
; IMPLICIT INPUTS:
;       @SCH$GL_PCBVEC  - VECTOR OF PCB ADDRESSES
;       PHD$L_PRIV      - PRIVILEGE BIT VECTOR IN PROCESS HEADER
;
; OUTPUT PARAMETERS:
;
;     If R0 EQL SS$_REMOTE_PROC:
;       R0        - SS$_REMOTE_PROC
;       R1        - EPID of process if PID <> 0, 0 otherwise
;       R2        - CSID of remote node
;       R3        - Modified, used for scratch register
;       R4        - PCB address of current process
;       IPL       - 0
;
;     If R0 EQL SS$_NORMAL
;       R0        - SS$_NORMAL
;       R1        - Internal process identification (IPID) of named process
;       R2,R3     - Modified, used for scratch registers
;       R4        - PCB address of target process
;       @PIDADR   - Extended process id (EPID) of selected process if
;                   PIDADR(AP) <> 0
```

```
738 ;       IPL      - IPL$_SYNCH
739 ;
740 ;     If R0 is anything else:
741 ;       R0       - Status code
742 ;       R1-R4    - Undefined
743 ;       @PIDADR  - Undefined (might have been written!)
744 ;       IPL      - IPL$_SYNCH (IPL unchanged if SS$_ACCVIO or SS$_IVLOGNAM)
745 ;
746 ; COMPLETION CODES:
747 ;       SS$_NORMAL -  NORMAL SUCCESSFUL COMPLETION
748 ;                           (R1 CONTAINS IPID)
749 ;       SS$_REMOTE_PROC -  EPID POINTS AT VALID REMOTE NODE
750 ;                           (R2 CONTAINS CSID)
751 ;       SS$_IVLOGNAM  -  INVALID LOGICAL NAME STRING
752 ;       SS$_NONEXPR -  NONEXISTENT PROCESS OR INVALID PID
753 ;       SS$_NOPRIV  -  NO PRIVILEGE FOR SPECIFIED OPERATION.
754 ;       SS$_ACCVIO  -  ACCESS VIOLATION FOR WRITE DESTINATION
755 ;
756 ; SIDE EFFECTS:
757 ;       NONE
758 ;
759 ;--
760
761
762 ; A VERY HIGH PROPORTION OF CALLS TO PROCESS SYSTEM SERVICES ARE FOR
763 ; THE CURRENT PROCESS.  WE CAN SIMPLIFY AND SPEED THAT PATH THROUGH
764 ; EXE$NAMPID, SINCE THERE IS VERY LITTLE TO CHECK OR DO...
765
766 CURRENT_PROCESS:
767         TSTL    R0                      ; WAS PID ADDRESS SPECIFIED?
768         BEQL    10$                     ; NO, SKIP STORE OF PID
769         MOVL    PCB$L_EPID(R4),(R0)     ; STORE EXTENDED PID IN DESTINATION
770 10$:    LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE (IPL$_SYNCH)
771                 PRESERVE=NO             ; R0 NO LONGER INTERESTING
772         MOVL    PCB$L_PID(R4),R1        ; USE CALLER'S PID
773         MOVZWL  #SS$_NORMAL,R0          ; SET SUCCESS STATUS
774         RSB                             ; AND RETURN TO CALLER
775
776 NP_ACCVIO:                             ; ACCESS VIOLATION
777         MOVZWL  #SS$_ACCVIO,R0          ; SET ERROR CODE
778         RSB                             ; AND EXIT
779
780 IVLNAM:                                ; INVALID NAME
781         MOVZWL  #SS$_IVLOGNAM,R0        ; SET ERROR CODE
782         RSB                             ; AND RETURN
783
784
785         UNIVERSAL_SYMBOL        EXE$NAMPID
786
787 ;EXE$NAMPID::                           ; TRANSLATE PNAME TO PID
788         CHECK_IPL 0
789         MOVL    PIDADR(AP),R0           ; GET PID ADDRESS
790         BEQL    130$                    ; NO PID ADDRESS
791         IFNOWRT #4,(R0),NP_ACCVIO       ; ERROR IF ACCESS VIOLATION
792         MOVL    (R0),R1                 ; NOW FETCH (EXTENDED) PID
793         BEQL    130$                    ; BRANCH IF NO PID FOUND
794         BSBW    EPID_TO_IPID            ; CONVERT EPID IN R1 TO IPID (LOCAL ROUTINE)
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 18
X-14 G^EXE$NAMPID - CONVERT PROCESS NAME TO P 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (

  795                                           ;  RETURNS AS IF COND CODES SET BY TSTL R3
  796          BEQL    GOTPID                   ; IF R3=0, NODE IS LOCAL, R1 IS THE IPID
  797          IFNOCLSTR GOTPID                 ; IF NOT CLUSTER, DO MAIN LINE CODE
  798          BRW     CWPS$CHECK_NODE          ; IT ISN'T LOCAL, GOTO CLUSTER CODE
  799
  800 130$:    MOVL    PRCNAM(AP),R3            ; GET PNAME ADDRESS IF SPECIFIED
  801          BEQL    CURRENT_PROCESS          ; NONE SPECIFIED, USE THE QUICK PATH
  802                                           ;  FOR THE CURRENT PROCESS
  803
  804 ; MUST LOOK UP PROCESS NAME, CHECK DESCRIPTOR AND NAME, THEN CALL THE
  805 ; LOOKUP ROUTINE
  806
  807          IFNORD  #8,(R3),NP_ACCVIO        ; CHECK DESCRIPTOR FOR READABLITY
  808          MOVQ    (R3),R2                  ; GET DESCRIPTOR
  809          TSTW    R2                       ; AND CHECK FOR ZERO LENGTH
  810          BEQL    IVLNAM                   ; NOT A VALID NAME STRING
  811          CMPW    #255,R2                  ; CHECK FOR MAXIMUM LENGTH
  812          BLSSU   IVLNAM                   ; NOT A VALID NAME STRING
  813          IFNORD  R2,(R3),NP_ACCVIO        ; ACCESS VIOLATION IF STRING NOT READABLE
  814
  815 ; R0 -> ADDRESS TO STORE PID (OR = 0)
  816 ; R2 =  LENGTH OF THE PROCESS NAME
  817 ; R3 -> THE PROCESS NAME TEXT
  818
  819          PUSHL   R0                       ; SAVE PID ADDRESS
  820 LCLNAM: MOVL     G^SCH$GL_MAXPIX,R0       ; INITIALIZE PROCESS INDEX
  821                                           ; LOOP FOR EACH PROCESS INDEX
  822 10$:     MOVL    @W^SCH$GL_PCBVEC[R0],R1  ; GET PCB ADDRESS FROM VECTOR
  823          CMPW    PCB$W_GRP(R1),PCB$W_GRP(R4) ; COMPARE GROUP NUMBERS
  824          BNEQ    20$                      ; NOT SAME GROUP, NEXT PIX
  825          CMPB    R2,PCB$T_LNAME(R1)       ; COMPARE NAME LENGTH
  826          BNEQ    20$                      ; DIFFERENT LENGTH
  827          PUSHR   #^M<R0,R1,R2,R3>         ; SAVE REGISTERS FOR CMPC3
  828          CMPC3   R2,(R3),PCB$T_LNAME+1(R1) ; COMPARE TEXT OF NAME
  829          POPR    #^M<R0,R1,R2,R3>         ; RESTORE REGISTERS
  830          BEQL    GOTNAM                   ; FOUND A MATCHING PROCESS NAME
  831 20$:     SOBGEQ  R0,10$                   ; UPDATE INDEX AND TRY AGAIN
  832
  833          BSBW    CWPS$PARSE_PRCNAM        ; CHECK FOR REMOTE PROCESS NAME
  834          BLBS    R0, LCLNAM               ; IF LBS, THEN THE NODENAME OF THE
  835                                           ;  LOCAL NODE WAS FOUND -- RETRY THE
  836                                           ;  SCAN (R2/R3 MOVED TO REMOVE NODENAME)
  837          ADDL2   #4, SP                   ; CLEAN PID ADDRESS FROM STACK
  838          RSB                              ; EXIT, CORRECT STATUS IN R0, AND
  839                                           ;  REGISTERS SET IF REMOTE PROC
  840
  841 GOTNAM: MOVL     PCB$L_PID(R1),R1         ; GET FULL PID FOR NAME
  842          POPL    R0                       ; RESTORE PID ADDRESS
  843 GOTPID:                                   ; VERIFY PID AND CHECK PRIV
  844          LOCK    LOCKNAME=SCHED,-         ; LOCK SCHED DATABASE
  845                  PRESERVE=YES             ; R0 HAS THE PID ADDRESS
  846          MOVZWL  R1,R2                    ; EXTRACT PROCESS INDEX
  847          CMPL    R2,G^SCH$GL_MAXPIX       ; TEST AGAINST MAXIMUM VALUE
  848          BGTRU   NONEX_UNLOCK             ; NONEXISTENT IF GTRU THAN MAXPIX
  849          MOVL    @W^SCH$GL_PCBVEC[R2],R2  ; GET PCB ADDRESS
  850          CMPL    R1,PCB$L_PID(R2)         ; CHECK FOR VALID PID
  851          BEQL    VALPID                   ; YES,
```

```
852 NONEX_UNLOCK:
853          UNLOCK  LOCKNAME=SCHED,-          ; UNLOCK SCHED DATABASE
854                  NEWIPL=#IPL$_ASTDEL,-     ; DROP IPL
855                  PRESERVE=NO               ; DON'T PRESERVE R0
856 NONEX:                                     ; PROCESS NON-EXISTENT
857          MOVZWL  #SS$_NONEXPR,R0           ; SET ERROR STATUS
858          RSB                               ; AND RETURN TO CALLER
859 VALPID:                                    ; PID IS VALID, CHECK PRIV
860          BSBB    EXE$CHECK_PCB_PRIV        ; CHECK PRIVS
861          BNEQ    NOPRIV                    ; IF COND CODE IS NEQ, NO PRIV
862 RETURN:                                    ; SUCCESSFUL EXIT
863          MOVL    R2,R4                     ; MOVE PCB ADDRESS OF TARGET
864                                            ; NORMAL STATUS EXIT
865          TSTL    R0                        ; WAS PID ADDRESS SPECIFIED
866          BEQL    10$                       ; NO, SKIP STORE OF PID
867          UNLOCK  LOCKNAME=SCHED,-          ; UNLOCK SCHED DATABASE
868                  NEWIPL=#IPL$_ASTDEL       ; DROP IPL
869          MOVL    PCB$L_EPID(R4),(R0)       ; STORE EXTENDED PID IN DESTINATION
870          CLRL    R0                        ; DO NOT WRITE PID A SECOND TIME
871          BRB     GOTPID                    ; MAKE SURE THAT PID IS STILL VALID
872
873 10$:     MOVZWL  #SS$_NORMAL,R0            ; SET SUCCESS STATUS
874          RSB                               ; AND RETURN TO CALLER
875
876 NOPRIV: UNLOCK   LOCKNAME=SCHED,-          ; UNLOCK SCHED DATABASE
877                  NEWIPL=#IPL$_ASTDEL,-     ; DROP IPL
878                  PRESERVE=NO               ; DON'T PRESERVE R0
879          MOVZWL  #SS$_NOPRIV,R0            ; SET ERROR STATUS
880          RSB                               ; AND RETURN TO CALLER
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 20
X-14 EXE$CHECK_PCB_PRIV - Check ability of on 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (

                        .SBTTL  EXE$CHECK_PCB_PRIV      - Check ability of one process to affect ano
                ;++
                ;       EXE$CHECK_PCB_PRIV
                ;
                ; FUNCTIONAL DESCRIPTION:
                ;       This routine checks that a given process (PCB address in
                ;       R4) has the privileges to examine or modify another process
                ;       (whose PCB address is in R2).
                ;
                ;
                ; CALLING SEQUENCE:
                ;       BSBx    EXE$CHECK_PCB_PRIV
                ;       BNEQ    NO_PRIV
                ;
                ; INPUT PARAMETERS:
                ;       R2 -> PCB of the target process
                ;       R4 -> PCB of the requestor process
                ;
                ; IMPLICIT INPUTS:
                ;       none
                ;
                ; OUTPUT PARAMETERS:
                ;       Condition codes set:
                ;               Z-bit set       - Process has enough priv
                ;               Z-bit clear     - Process does not have priv
                ;
                ; SIDE EFFECTS:
                ;       Z-bit modified, all other registers preserved
                ;
                ;--

                        UNIVERSAL_SYMBOL        EXE$CHECK_PCB_PRIV

                ;EXE$CHECK_PCB_PRIV::
                        CMPL    PCB$L_JIB(R2), PCB$L_JIB(R4)    ; Is it in our job (tree)?
                        BEQL    10$                             ; If so, allow it without privileges
                        CMPL    PCB$L_UIC(R2), PCB$L_UIC(R4)    ; Does process have same UIC?
                        BEQL    10$                             ; If so, allow it without privileges
                        IFPRIV  WORLD, 10$, R4                  ; Success if WORLD privilege
                        CMPW    PCB$W_GRP(R2), PCB$W_GRP(R4)    ; Are group numbers equal?
                        BNEQ    20$                             ; If not, no privilege
                        IFNPRIV GROUP, 20$, R4                  ; Error if not group priv

                10$:    BISPSW  #PSL$M_Z                        ; Set Z-bit   (equal)
                        RSB

                20$:    BICPSW  #PSL$M_Z                        ; Clear Z-bit (not equal)
                        RSB
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 21
X-14 CWPS$CHECK_NODE - check epid for valid r 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (

        .SBTTL  CWPS$CHECK_NODE - check epid for valid remote node
;+
;       A pid conversion has failed because the node fields of the pid are
;       non-zero and not the local node.  See if the node fields can be
;       expanded into a currently valid CSID.
;
; CALLING SEQUENCE:
;       BRW     CWPS$CHECK_NODE    - called from EXE$NAMPID when
;                                    non-local node is noticed
;
; INPUT PARAMETERS:
;       R2          - EXTENDED PID
;       R4          - PCB ADDRESS OF CURRENT PROCESS
;       PIDADR(AP) - ADDRESS OF PID SOURCE/DESTINATION (EXTENDED PID)
;       PRCNAM(AP) - POINTER TO PROCESS DESCRIPTOR TO CONVERT TO PID
;
; OUTPUT PARAMETERS (IF SS$_REMOTE_PROC ONLY, SEE EXE$NAMPID FOR SS$_NONEXPR)
;       R0        - SS$_REMOTE_PROC
;       R1        - EPID of process if PIDADR <> 0, 0 otherwise
;       R2        - CSID of remote node
;       R3        - Modified, used for scratch register
;       R4        - PCB address of current process
;       IPL       - 0
;
; COMPLETION CODES:
;       SS$_REMOTE_PROC -  EPID POINTS AT VALID REMOTE NODE
;       SS$_NONEXPR     -  NONEXISTENT PROCESS OR INVALID PID
;
; SIDE EFFECTS:
;       NONE
;
;--

NONEXB: UNLOCK  LOCKNAME=SCS,-          ; UNLOCK SCS DATABASE
                NEWIPL=#0,-             ;  DROP IPL
                PRESERVE=NO             ;  DON'T PRESERVE R0
        BRW     NONEX

CWPS$CHECK_NODE::

        CHECK_IPL 0

        LOCK    LOCKNAME=SCS, -         ; LOCK SCS DATABASE
                PRESERVE=NO             ;  DON'T PRESERVE R0
        MOVL    R2, R1                  ; PUT THE EPID WHERE WE WILL LEAVE IT
        EXTZV   #PCB$V_EPID_NODE_IDX,-  ; EXTRACT NODE INDEX TO R0
                #PCB$S_EPID_NODE_IDX,R1,R0
        EXTZV   #PCB$V_EPID_NODE_SEQ,-  ; EXTRACT NODE SEQUENCE NUMBER BITS
                #PCB$S_EPID_NODE_SEQ,R1,R2 ;  FROM THE EPID, ZERO EXTENDED
        CMPW    R0,G^CLU$GW_MAXINDEX    ; TEST AGAINST MAXIMUM VALUE
        BGEQU   NONEXB                  ; NONEXISTENT IF GEQU THAN MAXINDEX
        MOVL    G^CLU$GL_CLUSVEC,R3     ; GET POINT TO CSID VECTOR
        MOVL    (R3)[R0],R3             ; USE NODE INDEX TO GET CSB ADDRESS
        BGEQ    NONEXB                  ; GEQ MEANS IT'S UNUSED
        CMPZV   #0,-                    ; COMPARE THE LOW ORDER BITS OF THE
                #PCB$S_EPID_NODE_SEQ,-  ;  REAL SEQUENCE NUMBER FROM THE
                CSB$W_CSID_SEQ(R3),R2   ;  FULL CSID WITH EPID LOW ORDER BITS
```

```
  989          BNEQ    NONEXB                  ; NOT THE SAME
  990          MOVL    CSB$L_CSID(R3),R2       ; MOVE FULL CSID TO R2, R1 STILL EPID
  991          UNLOCK  LOCKNAME=SCS,-          ; UNLOCK SCS DATABASE
  992                  NEWIPL=#0,-             ;  LOWER IPL
  993                  PRESERVE=NO             ; DON'T PRESERVE R0
  994          MOVZWL  #SS$_REMOTE_PROC,R0     ; STATUS FOR GOOD NODE
  995          RSB
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 23
X-14 EXE$xPID_TO_xxx - CONVERT PID TO OTHER P 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (

  997          .SBTTL  EXE$xPID_TO_xxx - CONVERT PID TO OTHER PID OR PCB ADDRESS
  998 ;++
  999 ; FUNCTIONAL DESCRIPTIONS:
 1000 ;
 1001 ;       G^EXE$IPID_TO_PCB               - convert internal pid to pcb address
 1002 ;       G^EXE$EPID_TO_PCB               - convert extended pid to pcb address
 1003 ;       G^EXE$IPID_TO_EPID      - convert internal pid to extended pid
 1004 ;       G^EXE$EPID_TO_IPID      - convert extended pid to internal pid
 1005 ;
 1006 ; CALLING SEQUENCE:
 1007 ;       JSB/BSB EXE$xPID_TO_xxx
 1008 ;
 1009 ; INPUT PARAMETERS:
 1010 ;       R0      - input pid
 1011 ;
 1012 ; IMPLICIT INPUTS:
 1013 ;       @SCH$GL_PCBVEC - VECTOR OF PCB ADDRESSES
 1014 ;       G^SCH$GL_PIXWIDTH - WIDTH OF PIX FIELD IN EXTENDED PID
 1015 ;
 1016 ; OUTPUT PARAMETERS:
 1017 ;       R0      - output pid or pcb address, 0 if any problems
 1018 ;       CONDITION CODES - set according to the value in R0, so that any call
 1019 ;                         can be followed by a BEQL without another test
 1020 ;
 1021 ; COMPLETION CODES:
 1022 ;       NONE
 1023 ;
 1024 ; SIDE EFFECTS:
 1025 ;
 1026 ;       Non-paged code and data, no page faults possible.
 1027 ;
 1028 ;       Callers of these routines must be prepared for the routines to save
 1029 ;       registers R1 through R5 to allow for future additions.  For example,
 1030 ;       a BLISS linkage declaration of
 1031 ;
 1032 ;               LINKAGE
 1033 ;                   pid_call = JSB (REGISTER=0) : PRESERVE (1,2,3,4,5)
 1034 ;                                                 NOTUSED (6,7,8,9,10,11);
 1035 ;
 1036 ;       will force the enclosing procedure to save R2-R5 in the procedure
 1037 ;       entry mask.
 1038 ;--
 1039
 1040 ;+
 1041 ; Convert an extended PID to a PCB address.  We will first convert the EPID to an
 1042 ; IPID, then convert the IPID to the PCB address.  The condition codes will be set
 1043 ; according to the value in R0.
 1044 ;-
 1045          UNIVERSAL_SYMBOL        EXE$EPID_TO_PCB
 1046
 1047 ;EXE$EPID_TO_PCB::              ; CONVERT EXTENDED PID TO PCB ADDRESS
 1048          BSBB    EXE$EPID_TO_IPID        ; GET THE IPID IN R0
 1049          BEQL    10$                     ; COULDN'T CONVERT THE EPID
 1050          BSBB    EXE$IPID_TO_PCB         ; CONVERT THE IPID TO THE PCB ADDR
 1051 10$:    RSB
 1052
 1053 ;+
```

```
1054 ; Convert internal PID to PCB address.  Return 0 if the input IPID does not match
1055 ; the IPID stored in the corresponding PCB.  Set the condition codes according to
1056 ; the presence of a returned address in R0, so that the BSBx can be followed by a
1057 ; BEQL or BNEQ
1058 ;-
1059         UNIVERSAL_SYMBOL        EXE$IPID_TO_PCB
1060 ;EXE$IPID_TO_PCB::              ; CONVERT INTERNAL PID TO PCB ADDRESS
1061         CMPW    R0,G^SCH$GL_MAXPIX      ; TEST AGAINST MAXIMUM VALUE
1062         BGTRU   10$                     ; NONEXISTENT IF GTRU THAN MAXPIX
1063         PUSHL   R0                      ; SAVE A COPY OF THE IPID
1064         MOVZWL  R0,R0                   ; EXTRACT PROCESS INDEX FIELD
1065         MOVL    @W^SCH$GL_PCBVEC[R0],R0 ; MOVE PCB ADDRESS TO R0
1066         CMPL    PCB$L_PID(R0),(SP)+     ; DOES THE PID IN THE PCB MATCH?
1067         BNEQ    10$                     ; NO MATCH, RETURN 0 ADDRESS
1068         TSTL    R0                      ; SET THE CONDITION CODES
1069         RSB
1070 10$:    CLRL    R0                      ; NONEXISTENT PID, RETURN ZERO
1071         RSB
1072
1073 ;+
1074 ; Convert an extended PID to the internal PID.  Return 0 if the EPID refers to
1075 ; another node.  Do not check that either the EPID or IPID are valid.
1076 ;-
1077         UNIVERSAL_SYMBOL        EXE$EPID_TO_IPID
1078
1079 ;EXE$EPID_TO_IPID::             ; CONVERT EXTENDED PID TO INTERNAL PID
1080         PUSHR   #^M<R1,R2,R3>           ; SAVE SOME WORKING REGISTERS
1081         MOVL    R0, R1                  ; MOVE EPID TO R1
1082         BSBB    EPID_TO_IPID            ; CALL THE ROUTINE
1083         MOVL    R1, R0                  ; MOVE PID TO R0, SET COND CODES
1084         POPR    #^M<R1,R2,R3>           ; RESTORE REGISTERS
1085         RSB                             ; CONDITION CODES REFLECT NEW IPID
1086
1087 ;+
1088 ; Internal routine, convert an extended PID in R1 to the internal PID in R1.
1089 ; Return R3=1 and R1=0 if the EPID refers to another node.  Do not check if
1090 ; either the EPID or IPID is valid.  Leave condition codes set according to
1091 ; value in R3.  This routine assumes EXE$NAMPID's register usage.
1092 ;
1093 ;       R3      CC      R1 (IPID)       R2 (EPID)
1094 ;       ----    -----   ----------      ---------
1095 ;       0       EQL     IPID            ??
1096 ;       1       NEQ     0               EPID
1097 ;-
1098
1099 EPID_TO_IPID:                   ; CONVERT EXTENDED PID TO INTERNAL PID
1100
1101
1102 ; WE WILL EXTRACT THE NODE FIELD FROM THE EPID TO SEE IF THIS IS FOR THE LOCAL
1103 ; NODE.  WE WILL INCLUDE THE WILDCARD BIT IN THIS TEST.  VERIFY SOME ASSUMPTIONS
1104 ; ABOUT THE LOCATIONS OF THESE FIELDS.
1105
1106 NODE_WIDTH = PCB$S_EPID_NODE_IDX+PCB$S_EPID_NODE_SEQ
1107
1108         ASSUME  PCB$V_EPID_WILD EQ -            ; CHECK THAT WILD BIT IS RIGHT
1109                 <PCB$V_EPID_NODE_IDX + NODE_WIDTH> ;  AFTER NODE FIELDS
1110         ASSUME  PCB$V_EPID_NODE_SEQ EQ -        ;  AND SEQ IS RIGHT AFTER IDX
```

```
1111                    <PCB$V_EPID_NODE_IDX + PCB$S_EPID_NODE_IDX>
1112
1113            EXTZV   #PCB$V_EPID_NODE_IDX, - ; MOVE NODE + WILD TO R3
1114                    #<NODE_WIDTH+1>,R1,R3
1115            CMPW    G^SCH$GW_LOCALNODE,R3   ; IS IT THE LOCAL NODE?
1116            BNEQ    20$                     ; NOT LOCAL, CAN'T MAKE AN IPID
1117
1118 ; EPID IN R1 IS FOR LOCAL NODE, EXTRACT THE PIX AND SEQUENCE NUMBER TO FORM IPID
1119
1120 10$:      MOVL    G^SCH$GL_PIXWIDTH,R3    ; LOAD WIDTH OF EXTENDED PIX FIELD
1121            SUBL3   R3,#PCB$S_EPID_PROC,R2  ;  AND WIDTH OF THE SEQ NUM FIELD
1122            EXTZV   R3,R2,R1,R2             ; R2 IS LONGWORD SEQ NUM
1123            EXTZV   #0,R3,R1,R1             ; R1 IS LONGWORD PIX
1124            INSV    R2,#16,#15,R1           ; INSERT SEQ NUM IN HIGH WORD
1125                                            ;  WHICH MAKES AN IPID IN R1
1126            CLRL    R3                      ;
1127            RSB                             ; CONDITION CODES SET FOR VALUE OF R3
1128
1129 ; COULD NOT TURN EPID INTO AN IPID, RETURN AN IPID OF 0, WITH R3<>0, R2=EPID
1130
1131 20$:      MOVL    R1, R2                  ; PUT THE EPID IN R2
1132            CLRL    R1                      ; IPID = 0
1133            MOVL    #1, R3                  ; RETURN R3=1 AND COND CODE <> 0
1134            RSB
1135
1136
1137 ;+
1138 ; Convert an IPID to an EPID.  We do not check that the IPID is valid.  The local
1139 ; node is moved into the node field of the EPID, the seq number and pix of the IPID
1140 ; are moved into the EPID.  The condition codes reflect the final value of R0.
1141 ;-
1142            UNIVERSAL_SYMBOL        EXE$IPID_TO_EPID
1143
1144 ;EXE$IPID_TO_EPID::                 ; INTERNAL PID TO EXTENDED PID
1145            TSTL    R0                      ; TREAT A ZERO PID AS A SPECIAL CASE
1146            BEQL    10$                     ; ZERO, WE DON'T TOUCH IT
1147            PUSHR   #^M<R1,R2,R3>           ; SAVE SOME WORKING REGISTERS
1148            MOVZWL  R0,R3                   ; R3 IS LONGWORD PIX
1149            ASHL    #-16,R0,R0              ; R0 IS LONGWORD SEQ NUM
1150            MOVL    G^SCH$GL_PIXWIDTH,R1    ; LOAD WIDTH OF EXTENDED PIX FIELD
1151            SUBL3   R1,#PCB$S_EPID_PROC,R2  ; AND THE WIDTH OF THE SEQ NUM FIELD
1152            INSV    R0,R1,R2,R3             ; INSERT SEQ NUM BESIDE PIX
1153            INSV    G^SCH$GW_LOCALNODE, -   ; INSERT LOCAL NODE INTO THE EPID
1154                    #PCB$V_EPID_NODE_IDX, #NODE_WIDTH, R3
1155            MOVL    R3,R0                   ; RETURN THE EPID IN R0
1156            POPR    #^M<R1,R2,R3>           ; RESTORE REGISTERS
1157 10$:      RSB                             ; N.B. COND CODES SET ON VALUE OF R0
1158
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 26
X-14 G^EXE$SETPRN - SET PROCESS NAME 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (12)

1160          .SBTTL  G^EXE$SETPRN - SET PROCESS NAME
1161
1162 ;++
1163 ; FUNCTIONAL DESCRIPTION:
1164 ;       G^EXE$SETPRN IMPLEMENTS THE SET PROCESS NAME SYSTEM
1165 ;       SERVICE WHICH ALLOWS A PROCESS TO ESTABLISH A LOGICAL NAME
1166 ;       FOR ITSELF.  ALL SUCH LOGICAL NAMES ARE IMPLICITLY QUALIFIED
1167 ;       BY THE GROUP NUMBER OF THE PROCESS THEREBY ALLOWING THE SAME
1168 ;       LOGICAL NAME TO BE USED BY PROCESSES IN DIFFERENT GROUPS.
1169 ;
1170 ; CALLING SEQUENCE:
1171 ;       CALLG   ARGLIST,G^EXE$SETPRN
1172 ;
1173 ; INPUT PARAMETERS:
1174 ;       04(AP) - ADDRESS OF PROCESS NAME STRING DESCRIPTOR
1175 PRCNAM=4
1176 ;       R4 - PCB ADDRESS OF CURRENT PROCESS
1177 ;
1178 ; IMPLICIT INPUTS:
1179 ;       CPU$L_CURPCB in per-cpu data - POINTER TO PCB OF CURRENT PROCESS
1180 ;       @SCH$GL_PCBVEC - VECTOR OF ALL PCB ADDRESSES
1181 ;
1182 ; OUTPUT PARAMETERS:
1183 ;       NONE
1184 ;
1185 ; IMPLICIT OUTPUTS:
1186 ;       PCB$T_NAME IN CURRENT PCB IS FILLED WITH THE SPECIFIED NAME
1187 ;       PROVIDED NO ERROR HAS OCURRED.
1188 ;
1189 ; SIDE EFFECTS:
1190 ;       NONE
1191 ;
1192 ; COMPLETION CODES:
1193 ;       SS$_NORMAL     -  NORMAL SUCCESSFUL COMPLETION STATUS
1194 ;       SS$_ACCVIO     -  ALL OR PART OF NAME STRING IS INACCESSIBLE FOR READ
1195 ;       SS$_IVLOGNAM   -  ILLEGAL LOGICAL NAME STRING LENGTH (>15)
1196 ;       SS$_DUPLNAM    -  DUPLICATE PROCESS NAME WITHIN GROUP
1197 ;
1198 ;--
1199
1200          SYSTEM_SERVICE  SETPRN,-
1201                          <R2,R3,R4,R5,R6,R7,R8,R9>,-
1202                          MODE=KERNEL,-
1203                          NARG=1
1204
1205          MOVL    PRCNAM(AP),R5          ; GET ADDRESS OF PROCESS NAME
1206          BNEQ    100$                   ; WAS SPECIFIED
1207          CLRL    PCB$T_LNAME(R4)        ; CLEAR NAME FIELD OF PCB
1208          BRB     170$                   ; AND EXIT WITH NORMAL STATUS
1209 100$:    IFNORD  #8,(R5),190$           ; CHECK ACCESS FOR DESCRIPTOR
1210          MOVQ    (R5),-(SP)             ; PUSH DESCRIPTOR ON STACK
1211          TSTW    (SP)                   ; CHECK FOR ZERO LENGTH STRING
1212          BEQL    110$                   ; INVALID NAME
1213          IFNORD  (SP),@4(SP),190$       ; PROBE ENDS OF STRING
1214          CMPW    (SP),#15               ; CHECK FOR MAXIMUM LENGTH
1215          BLEQU   120$                   ; IF LEQU, WITHIN LIMIT
1216 110$:    MOVZWL  #SS$_IVLOGNAM,R0       ; INVALID PROCESS NAME STATUS
```

```
1217          RET                             ; AND RETURN
1218  120$:   MOVL    G^SCH$GL_MAXPIX,R6       ; SET MAXIMUM PROCESS INDEX
1219  130$:   MOVL    @W^SCH$GL_PCBVEC[R6],R7  ; GET PCB ADDRESS
1220          CMPW    PCB$W_GRP(R4),PCB$W_GRP(R7)  ; CHECK FOR SAME GROUP
1221          BNEQ    140$                     ; NO, SKIP IT
1222          CMPB    (SP),PCB$T_LNAME(R7)     ; COMPARE LENGTHS
1223          BNEQ    140$                     ; NOT EQUAL, TRY ANOTHER
1224          CMPC3   (SP),@4(SP),PCB$T_LNAME+1(R7)    ; COMPARE NAMES WITH COUNTS
1225          BEQL    150$                     ; MATCH
1226  140$:   SOBGEQ  R6,130$                  ; CONTINUE FOR ALL PCBS
1227          BRB     160$                     ; NOT FOUND
1228  150$:   CMPL    R4,R7                    ; SAME PROCESS?
1229          BNEQ    180$                     ; DUPLICATE NAME ERROR
1230  160$:   MOVB    (SP),PCB$T_LNAME(R4)     ; SAVE NAME LENGTH
1231          MOVC3   (SP),@4(SP),PCB$T_LNAME+1(R4)    ; MOVE NAME TO PCB
1232  170$:   MOVZWL  #SS$_NORMAL,R0           ; SUCCESSFUL STATUS
1233          RET                              ; AND RETURN
1234  180$:   MOVZWL  #SS$_DUPLNAM,R0          ; DUPLICATE NAME WITHIN GROUP
1235          RET                              ; AND RETURN
1236
1237  190$:   MOVZWL  #SS$_ACCVIO,R0           ; ACCESS VIOLATION
1238          RET                              ; RETURN WITH ERROR STATUS
1239
```

```
1241          .SBTTL  CWPS$PARSE_PRCNAM - CWPS PARSE PROCESS NAME
1242 ;++
1243 ;       CWPS$PARSE_PRCNAM  -  CWPS PARSE PROCESS NAME
1244 ;
1245 ; FUNCTIONAL DESCRIPTION:
1246 ;
1247 ;       CWPS$PARSE_PRCNAM separates a process name into its component parts.
1248 ;       A fully qualified process name string is
1249 ;
1250 ;               _NODE::PROCNAM  or  _NODE::PROCNAM
1251 ;
1252 ;       WHERE
1253 ;               _NODE::         is the node name of the cluster node where the
1254 ;                                  process is located
1255 ;               PROCNAM         1-15 character process name as stored in the pcb
1256 ;
1257 ;       The node field only will be subjected to logical name translation.
1258 ;
1259 ; CALLING SEQUENCE:
1260 ;       JSB/BSB CWPS$PARSE_PRCNAM
1261 ;
1262 ; INPUT PARAMETERS:
1263 ;       R4 -> PCB of current process
1264 ;       R3 -> PROCESS NAME DESCRIPTOR, DESC and STRING have been probed
1265 ;       R2 =  Length of name string.
1266 ;
1267 ; IMPLICIT INPUTS:
1268 ;       NONE
1269 ;
1270 ; OUTPUT PARAMETERS:
1271 ;
1272 ;       R0 - COMPLETION STATUS
1273 ;
1274 ;         IF R0 = SS$_NORMAL
1275 ;         THEN
1276 ;               R1 =  ?
1277 ;               R2 =  LENGTH OF NAME (NODENAME:: HAS BEEN REMOVED)
1278 ;               R3 -> NAME STRING (AFTER THE NODENAME::)
1279 ;               R4 -> CURRENT PCB
1280 ;
1281 ;         ELSE IF R0 = SS$_REMOTE_PROC  (A WARNING STATUS)
1282 ;         THEN
1283 ;               R1 =  0
1284 ;               R2 =  CLUSTER SYSTEM ID (CSID) OF THE REMOTE SYSTEM
1285 ;               R3 =  ?
1286 ;               R4 -> CURRENT PCB
1287 ;
1288 ;         ELSE
1289 ;               R0 CONTAINS THE ERROR CODE
1290 ;                       SS$_NONEXPR       - NONEXISTANT PROCESS (NO NODENAME
1291 ;                                           IN STRING
1292 ;                       SS$_IVLOGNAM      - INVALID LOGICAL NAME STRING
1293 ;                       SS$_ACCVIO        - ACCESS VIOLATION FOR READING STRING
1294 ;                       SS$_NOSUCHNODE    - STRING REFERS TO NONEXISTANT NODE
1295 ;               R1 =  ?
1296 ;               R2 =  ?
1297 ;               R3 =  ?
```

```
1298 ;               R4 -> CURRENT PCB
1299 ;
1300 ; SIDE EFFECTS:
1301 ;
1302 ;--
1303
1304         DECLARE_PSECT   EXEC$PAGED_CODE
1305
1306         Universal_symbol        CWPS$PARSE_PRCNAM
1307
1308 ;CWPS$PARSE_PRCNAM::
1309
1310         CHECK_IPL 0
1311
1312         IFNOCLSTR 10$                   ; IF NOT CLUSTER, WE ARE DONE EXCEPT
1313                                         ;  FOR DECIDING WHICH ERROR STATUS
1314
1315 ; SEE IF THERE IS A NODE SPEC IN THE STRING.  WE SIMPLY LOOK FOR THE ::
1316 ;
1317 ;       R2 =  LENGTH OF PROCESS NAME STRING
1318 ;       R3 -> STRING, PROBED FOR READING
1319
1320         LOCC    #^A:,R2,(R3)            ; SEARCH STRING FOR A COLON
1321         BEQL    10$                       ; IF EQL COLON NOT FOUND
1322         DECL    R0                        ; POSSIBLY A NODE NAME?
1323         BEQL    10$                       ; IF EQL NO
1324         CMPB    #^A:,1(R1)              ; NEXT CHARACTER A COLON?
1325         BEQL    PARSE_NODE                ; IF EQL, TURN NODE INTO CSID AND RETURN
1326
1327 ; NO NODENAME, AND WE'VE ALREADY CHECKED THE LOCAL NODE, THEREFORE THE ERROR
1328 ; IS EITHER NO SUCH PROCESS OR NAME TOO LONG
1329
1330 ; NOTE:  WE DO NOT CHECK THE LENGTH OF A REMOTE NAME, WE LEAVE THAT PROBLEM
1331 ;        TO THE REMOTE NODE.  IN THE EVENT THAT THE MAXIMUM LENGTH OF A PROCESS
1332 ;        NAME IS EXTENDED, THIS WILL PERMIT GOOD MIXED-VERSION OPERATION (AN
1333 ;        OLD NODE WILL BE ABLE TO REFERENCE A PROCESS ON A NEW NODE BY USING
1334 ;        A LONG NAME, EVEN IF THAT NAME WOULD BE INVALID LOCALLY).
1335
1336 10$:    MOVZWL  #SS$_NONEXPR, R0        ; SET ERROR CODE FOR NO SUCH PROC
1337         CMPW    #15, R2                 ; CHECK FOR MAXIMUM LENGTH NAME
1338         BGEQU   90$                     ; LENGTH IS OK, SS$_NONEXPR IS CORRECT
1339         MOVZWL  #SS$_IVLOGNAM, R0       ; LENGTH TOO LONG, SET ANOTHER ERROR
1340 90$:    RSB
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 30
X-14 PARSE_NODE - PARSE NODE NAME 7-APR-1989 13:14:24 [SYS.SRC]SYSPCNTRL.MAR;1 (14)

1342          .SBTTL  PARSE_NODE - PARSE NODE NAME
1343 ;++
1344 ;       PARSE_NODE  -  PARSE NODE NAME
1345 ;
1346 ; FUNCTIONAL DESCRIPTION:
1347 ;       PARSE_NODE CONVERTS A NODE STRING TO THE CLUSTER SYSTEM ID (CSID) FOR
1348 ;       THAT NODE.  THE NODE WILL BE SUBJECTED TO LOGICAL NAME TRANSLATIONS.
1349 ;
1350 ;       IF THE NODE IS THE LOCAL NODE, THE NODENAME IS REMOVED FROM THE STRING
1351 ;       BY ADJUSTING R2 AND R3 TO SKIP OVER THE NODENAME AND THE :: AND NO
1352 ;       CSID IS RETURNED.
1353 ;
1354 ; CALLING SEQUENCE:
1355 ;       JSB/BSB PARSE_NODE
1356 ;
1357 ; INPUT PARAMETERS:
1358 ;       R0 =  LENGTH OF THE PROCESS NAME, FROM THE SECOND COLON TO THE END
1359 ;       R1 -> FIRST OF TWO COLONS (::) SEPARATING NODE FROM THE REST OF NAME
1360 ;       R2 =  TOTAL LENGTH OF PROCESS NAME STRING
1361 ;       R3 -> BEGINNING OF PROCESS NAME STRING, PROBED FOR READING
1362 ;
1363 ; IMPLICIT INPUTS:
1364 ;       CLU$GL_CLUB     -> CLUB FOR LOCAL NODE
1365 ;       CLU$GL_CLUSVEC  -> VECTOR OF CSB'S DESCRIBING THE CLUSTER
1366 ;       CLU$GW_MAXINDEX =  SIZE OF VECTOR OF CSB'S
1367 ;       WE ARE IN A CLUSTER...
1368 ;
1369 ; OUTPUT PARAMETERS:
1370 ;
1371 ;         IF R0 IS SUCCESS
1372 ;               R1 =  ?
1373 ;               R2 =  LENGTH OF PROCESS NAME (WITHOUT THE NODENAME)
1374 ;               R3 -> PROCESS NAME, PROBED FOR READ (WITHOUT THE NODENAME)
1375 ;
1376 ;         ELSE IF R0 = SS$_REMOTE_PROC  (AN ERROR STATUS)
1377 ;               R1 =  0
1378 ;               R2 =  CLUSTER SYSTEM ID (CSID) OF THE REMOTE SYSTEM
1379 ;               R3 =  ?
1380 ;         ELSE
1381 ;               R0 CONTAINS THE ERROR CODE
1382 ;                       SS$_NOSUCHNODE  - STRING REFERS TO NONEXISTANT NODE
1383 ;                       SS$_IVLOGNAM    - NODE NAME IS TOO LONG
1384 ;
1385 ; SIDE EFFECTS:
1386 ;       R0, R1 AND R4 WILL BE TRASHED
1387 ;
1388 ;--
1389
1390          DECLARE_PSECT   EXEC$PAGED_CODE
1391
1392 PARSE_NODE:
1393          SAVE_MASK =^M<R4,R5,R6,R7,R8,R9>; DEFINE REGISTER SAVE MASK
1394          PUSHR   #SAVE_MASK              ; SAVE REGISTERS WHICH WE WILL USE
1395
1396 ; COMPUTE THE LENGTH OF THE NODE NAME, ERROR IF TOO LONG FOR A LOGICAL NAME.
1397 ; ADJUST THE STRING POINTERS FOR THE PROCESS NAME
1398
```

```
1399          SUBL3   R3,R1,R7                ; LENGTH OF NODE NAME TO R7
1400          CMPL    #LOG$C_NAMLENGTH-1,R7   ; COMPARE AGAINST MAX FOR LOGICAL NAME
1401          BLSS    120$                    ; INVALID NAME IF TOO LONG
1402          MOVL    R3,R8                   ; ADDRESS OF NODE NAME TO R8
1403          ADDL2   R7,R3                   ; MOVE PROCNAME POINTER PAST THE NODE
1404          ADDL2   #2,R3                   ; MOVE PROCNAME POINTER PAST THE ::
1405          SUBL3   #1,R0,R2                ; LENGTH IS ONE LESS THAN REMAINING
1406
1407 ; RECURSIVELY TRANSLATE THE NODENAME UNTIL SS$_NOTRAN OR DEFAULT NUMBER OF
1408 ; TRANSLATIONS.  PREPARE A TRANSLATE BUFFER ON THE STACK:
1409 ;
1410 ;         ---------------------------------
1411 ; SP:     |  Logical name length          |
1412 ;         |-------------------------------|
1413 ;  +4     |  Logical name address         |
1414 ;         |-------------------------------|
1415 ;  +8     |  Result buffer length         |
1416 ;         |-------------------------------|
1417 ;  +12    |  Result buffer address        |
1418 ;         |-------------------------------|
1419 ;  +16    |  logical name / result        |
1420 ;         |       buffer                  |
1421 ;         ---------------------------------
1422
1423 LNLEN = 0                               ; OFFSET TO LOGICAL NAME LENGTH
1424 LNADR = 4                               ;                 ADDRESS
1425 RBLEN = 8                               ;       RESULT BUFFER LENGTH
1426 RBADR = 12                              ;                 ADDRESS
1427 BUFF  = 16                              ; BUFFER FOR BOTH NAME AND RESULT BUFFER
1428 TOTAL = 16+LOG$C_NAMLENGTH              ; TOTAL SIZE OF TRANSLATE BUFFER
1429 ASSUME  LOG$C_NAMLENGTH LE 255          ; ASSUME THIS FITS IN A BYTE
1430
1431 ;       R8 -> NODE NAME STRING
1432 ;       R7 =  LENGTH OF THE NODE NAME
1433 ;       R6 =  (WILL BE SCRATCH POINTER)
1434 ;       R5 =  ?
1435 ;       R4 =  ?
1436 ;       R3 -> BEGINNING OF PROCESS NAME STRING, ADJUSTED PAST THE NODE NAME
1437 ;       R2 =  LENGTH OF PROCESS NAME STRING, AS ABOVE
1438
1439          MOVAB   -TOTAL(SP),SP           ; ALLOCATE STORAGE FOR BUFFER ON STACK
1440          MOVL    SP,R6                   ; SAVE THE BUFFER ADDRESS
1441          MOVAB   BUFF(R6),LNADR(R6)      ; SET LOGNAM BUFFER ADDRESS IN DESC
1442          MOVL    R7,LNLEN(R6)            ; SET LOGNAM LENGTH IN DESC
1443          MOVAB   BUFF(R6),RBADR(R6)      ; SET RESULT BUFFER ADDRESS IN DESC
1444          MOVQ    R2,-(SP)                ; PROTECT REGISTERS FROM MOVC
1445          MOVC3   R7,(R8),BUFF(R6)        ; MOVE THE NODE NAME TO THE BUFFER
1446          MOVQ    (SP)+,R2                ; RESTORE REGISTERS
1447          MOVL    #10,R7                  ; SET ITERATION COUNTER
1448 100$:   MOVZBL  #LOG$C_NAMLENGTH,-       ; SET RESULT BUFFER TO FULL LENGTH
1449                  RBLEN(R6)               ;  AT THE TOP OF EACH ITERATION
1450          $TRNLOG_S LOGNAM=LNLEN(R6),-    ; TRANSLATE LOGICAL NAME
1451                    RSLLEN=RBLEN(R6),-
1452                    RSLBUF=RBLEN(R6)
1453          BLBS    R0,110$                 ; SKIP OVER BUGCHECK IF IT WORKED
1454          BSBW    180$                    ; SHOULDN'T FAIL UNLESS SOMETHING IS BAD
1455 110$:   CMPW    #SS$_NOTRAN,R0          ; ARE WE DONE WITH TRANSLATIONS?
```

```
SYSPCNTRL PROCESS CONTROL SERVICES 10-MAY-1989 16:43:21 VAX MACRO V5.0-8 Page 32
X-14 PARSE_NODE - PARSE NODE NAME 7-APR-1989 13:14:34 [SYS.SRC]SYSPCNTRL.MAR;1 (14)

 1456          BEQL    130$                    ; YES, USE THE NODE NAME
 1457          MOVL    RBLEN(R6),LNLEN(R6)     ; RECYCLE OUTPUT AS INPUT NAME
 1458          SOBGTR  R7,100$                 ; DO IT FOR TEN TIMES BEFORE GIVING UP
 1459          MOVAB   TOTAL(SP),SP            ; CLEAN TRANSLATE BUFFER FROM STACK
 1460
 1461 ; INVALID LOGICAL NAME ERROR
 1462
 1463 120$:    MOVZWL  #SS$_IVLOGNAM,R0        ; SET ERROR CODE
 1464          BRW     160$                    ; AND EXIT
 1465
 1466 ; NODE HAS BEEN TRANSLATED, AND ANY LEADING _ HAS BEEN REMOVED.  FIRST SEE IF
 1467 ; THIS NODE IS THE LOCAL NODE
 1468 ;
 1469 ;        R8 =  ?
 1470 ;        R7 =  ?
 1471 ;        R6 -> TRANSLATION BUFFER CONTAINING THE NAME
 1472 ;        R5 =  ?
 1473 ;        R4 =  ?
 1474 ;        R3 -> BEGINNING OF PROCESS NAME STRING, ADJUSTED PAST THE NODE NAME
 1475 ;        R2 =  LENGTH OF PROCESS NAME STRING
 1476
 1477 130$:    MOVQ    RBLEN(R6),R7            ; MOVE NODENAME LENGTH TO R7, ADDRESS
 1478                                          ;  OF NODENAME TO R8
 1479
 1480 ; CONVERT NODE NAME TO UPPER CASE, AS SB NODE NAMES ARE ALWAYS UPPER CASE
 1481
 1482          MOVL    G^EXE$AR_UPCASE_DAT,R0  ; GET POINTER TO UPCASE TABLE
 1483          CLRL    R1                      ; CLEAR UPPER BYTES OF INDEX
 1484 140$:    MOVB    (R8),R1                 ; GET THE NEXT BYTE
 1485          MOVB    (R0)[R1],(R8)+          ; GET NEW CHAR, ADVANCE POINTER
 1486          SOBGTR  R7,140$                 ; LOOP THROUGH NODE NAME STRING
 1487
 1488 ; PREPARE TO SCAN CLUSTER NAME DATABASE
 1489
 1490          MOVQ    RBLEN(R6),R7            ; NODENAME LENGTH TO R7, ADDRESS TO R8
 1491          MOVL    G^CLU$GL_CLUB,R9        ; GET POINTER TO CLUSTER BLOCK
 1492          BEQL    180$                    ; IF NO CLUB, THEN NOT IN CLUSTER
 1493          PMLREQ  END=LOCK_END            ; LOCK CODE WHICH RUNS AT HIGH IPL

      ;++
      ; NB: Co-routine address + 2 LWs have been placed on top of stack
      ;--
 1494          LOCK    LOCKNAME=SCS, -         ; LOCK SCS DATABASE AND RAISE
 1495                  PRESERVE=NO             ;  IPL TO SYNCH
 1496          MOVL    CLUB$L_LOCAL_CSB(R9),R9 ; GET POINTER TO CSB FOR LOCAL SYSTEM
 1497          BEQL    180$                    ; IF NO CSB, THEN SOMETHING IS ROTTEN
 1498          MOVL    CSB$L_SB(R9),R1         ; R1 -> Local SB
 1499          BEQL    180$                    ; IF EQL NONE???
 1500          MOVQ    R2,-(SP)                ; SAVE REGISTERS FROM THE CMPC
 1501          MOVAB   SB$T_NODENAME(R1),R3    ; MOVE ADDRESS OF STRING (COUNT BYTE)
 1502          MOVZBL  (R3)+,R2                ; LENGTH TO R2, R3 -> TO TEXT FIELD
 1503          CMPC5   R2,(R3),#0,R7,(R8)      ; DOES THE INPUT NODE MATCH THE LOCAL?
 1504          BNEQ    190$                    ; NOT A MATCH ON LOCAL, CHECK ENTIRE CLUSTER
 1505          MOVQ    (SP)+,R2                ; RESTORE REGISTERS
 1506          UNLOCK  LOCKNAME=SCS,-          ; UNLOCK SCS DATABASE
 1507                  NEWIPL=#0,-             ;  LOWER IPL
 1508                  PRESERVE=NO             ;  R0 DOESN'T MATTER
```

```
1509          PMLEND                          ; UNLOCK PAGES

      ;++
      ; NB: Co-routine address + 2 LWs have been removed from top of stack
      ;--
1510          MOVZWL  #SS$_NORMAL, R0         ; STATUS SAYS ITS THE LOCAL NODE,
1511                                          ;  R2/R3 POINT TO PROCESS NAME ONLY
1512
1513 ; REMOVE THE TRANSLATE BUFFER FROM THE STACK
1514
1515 150$:    MOVAB   TOTAL(SP),SP            ; DISCARD THE TRANSLATE BUFFER
1516
1517 ; COMMON RETURN, RESTORE SAVED REGISTERS
1518
1519 160$:    POPR    #SAVE_MASK              ; RESTORE THE SAVED REGISTERS
1520          RSB
1521
1522 ; LOCAL SYSTEM IS NOT IN A CLUSTER, OR NODE SPECIFIED IS NOT IN THIS CLUSTER
1523
1524 170$:    MOVZWL  #SS$_NOSUCHNODE,R0      ; SET ERROR CODE
1525          BRB     150$                    ; GO DISCARD TRANSLATE BUFFER AND EXIT
1526
1527 180$:    BUG_CHECK  SSRVEXCEPT,FATAL     ; INCONSISTENT DATA
1528
1529 ; THE NODE IS NOT THE LOCAL NODE, CHECK ALL THE NODES IN THE CLUSTER
1530 ;
1531 ;        R9 =  (WILL BE CSID VECTOR INDEX)
1532 ;        R8 -> TRANSLATED NODE NAME
1533 ;        R7 =  LENGTH OF NODE NAME
1534 ;        R6 =  (WILL BE CLUSVEC POINTER)
1535 ;        R5 =  (WILL BE CSB ADDRESS)
1536 ;        R4 =  ?
1537 ;        R3 =  ?
1538 ;        R2 =  ?
1539 ;        SAVED R2/R3 ON TOP OF THE STACK
1540
1541 190$:    MOVZWL  G^CLU$GW_MAXINDEX,R9    ; GET COUNT OF ENTRIES IN CLUSTER VECTOR
1542          DECL    R9                      ; ADJUST, SINCE COUNT IS MAX INDX + ONE
1543          BLEQ    180$                    ; COUNT SHOULD BE POSITIVE
1544 200$:    MOVL    G^CLU$GL_CLUSVEC,R6     ; R6 -> CSID vector
1545 210$:    MOVL    (R6)[R9],R5             ; USE HI NODE INDEX TO GET CSB ADDRESS
1546          BGEQ    230$                    ; IF NOT SYSTEM ADDRESS, NOT A CSB
1547          MOVL    CSB$L_SB(R5),R1         ; R1 -> NODE'S SYSTEM BLOCK
1548          BEQL    180$                    ; IF EQL NO SB FOR THIS CSB???
1549 220$:    MOVAB   SB$T_NODENAME(R1),R3    ; MOVE ADDRESS OF STRING (COUNT BYTE)
1550          MOVZBL  (R3)+,R2                ; GET LENGTH IN R2, ADVANCE R3 TO TEXT FIELD
1551          CMPC5   R2,(R3),#0,R7,(R8)      ; DOES THE INPUT NODE MATCH THIS SB?
1552          BEQL    240$                    ; WE FOUND IT
1553 230$:    SOBGEQ  R9,210$                 ; LOOK AT EVERY ENTRY IN THE VECTOR
1554          UNLOCK LOCKNAME=SCS,-           ; UNLOCK SCS DATABASE
1555                 NEWIPL=#0,-              ;  LOWER IPL
1556                 PRESERVE=NO              ;  R0 IS NOT IMPORTANT
1557          MOVQ    (SP)+,R2                ; RESTORE REGISTERS
1558          PMLEND                          ; UNLOCK PAGES

      ;++
      ; NB: Co-routine address + 2 LWs have been removed from top of stack
```

```
                ;--
 1559
 1560 ; WE COULD NOT FIND THIS NODE, RETURN THE SS$_NOSUCHNODE ERROR
 1561
 1562                BRW     170$                    ; RETURN SS$_NOSUCHNODE AND EXIT
 1563
 1564 ; WE FOUND THE NODE, SET UP REGISTERS AND RETURN THE SS$_REMOTE_PROC VALUE
 1565 ;
 1566 ;       R5 -> CSB FOR THE LOCATED NODE
 1567 ;       <R2,R3> SAVED ON STACK, HAVE PROCNAME LENGTH AND ADDRESS
 1568 ;       <R0,R1,R4,R5,R7,R8,R9> NOT INTERESTING
 1569
 1570 240$:   MOVL    CSB$L_CSID(R5),R2       ; GET THE CSID FROM THE CSB
 1571         UNLOCK  LOCKNAME=SCS,-          ; UNLOCK SCS DATABASE
 1572                 NEWIPL=#0,-             ;  LOWER IPL
 1573                 PRESERVE=NO             ;  R0 IS NOT INTERESTING
 1574         ADDL2   #8,SP                   ; DISCARD SAVED REGISTERS
 1575         PMLEND                          ; UNLOCK PAGES
                ;++
                ; NB: Co-routine address + 2 LWs have been removed from top of stack
                ;--
 1576         CLRL    R1                      ; NO PID TO RETURN
 1577         MOVZWL  #SS$_REMOTE_PROC,R0     ; RETURN REMOTE PROCESS STATUS
 1578         BRW     150$                    ; CLEAN STACK AND RETURN
 1579
 1580 LOCK_END:
```

```
1582          .SBTTL  CWPS$PCNTRL     Cluster pcntrl dispatcher
1583 ;+
1584 ;
1585 ;       Routine to dispatch simple process request to another cluster
1586 ;       node, and to return status to the user.
1587 ;
1588 ; Calling sequence:
1589 ;       Branch from local service code
1590 ;
1591 ; Current state:
1592 ;       PIDADR(AP) - Address of EPID for target process (optional)
1593 ;       PRCNAM(AP) - Address of process name    (optional)
1594 ;       R0         - Status code SS$_REMOTE_PROC from EXE$NAMPID
1595 ;       R1         - EPID of remote process (or 0)
1596 ;       R2         - CSID of remote node
1597 ;       R3         - CWPS Service code
1598 ;       R4         - PCB address of current process
1599 ;       R5         - available
1600 ;       IPL        - IPL = 0
1601 ;
1602 ; Outputs from routine:  (Note - this routine JUMPs to cluster code which will
1603 ;                                perform the RET which returns us to the service
1604 ;                                caller.)
1605 ;       R0         - Completion status from remote node or CSP condition
1606 ;       (R5)       - If R5<>0, then receives return data
1607 ;
1608 ; Implicit output:
1609 ;      @PIDADR(AP) - If <>0, receives EPID of process actually modified.  This
1610 ;                    address is probed and written by the routine
1611 ;                    CWPS$SSND_PCNTRL_RQST
1612 ;
1613 ;-
1614
1615 CWPS$PCNTRL::
1616
1617          ; Allocate a CWPSSRV$ block.  This block contains pcntrol
1618          ; service-common fields. Point R0 at the block.
1619
1620     ASSUME cwpscan$k_length EQ 0        ; No extensions for these guys
1621     ASSUME cwpsdel$k_length EQ 0
1622     ASSUME cwpsres$k_length EQ 0
1623     ASSUME cwpssus$k_length EQ 0
1624     ASSUME cwpswak$k_length EQ 0
1625
1626          movq    R1, -(SP)                      ; Save R1:R2 across call
1627          clrl    R0                             ; No specific extension
1628          clrl    R1                             ; No additional buffers
1629          movab   PIDADR(AP), R2                 ; Pass address of pidadr/prcnam
1630          jsb     g^CWPS$ALLOCATE_SRV            ; Allocate a block (and other
1631                                                 ;  stuff, and some setups)
1632          ; IPL now is IPL$_ASTDEL
1633
1634          blbc    R0, 10$                        ; Exit if error
1635          movl    R2, R0                         ; Copy pointer to R0
1636          movq    (SP)+, R1                      ; Restore R1:R2
1637
1638          ; Now the block address is in R0, the size in CWPSSRV$L_SEND_LEN(R0)
```

```
1639          ; and CWPSSRV$L_MAXIMUM_LENGTH(R0).  Set the type/subtype fields.
1640
1641          ASSUME  cwpssrv$b_type      EQ  cwpssrv$w_size+2
1642          ASSUME  cwpssrv$b_subtype   EQ  cwpssrv$b_type+1
1643
1644          movb    #dyn$c_cwps, cwpssrv$b_type(R0) ; Set type code
1645          movb    R3, cwpssrv$b_subtype(R0)       ; Load subtype code
1646
1647          ; Set extension structure levels
1648
1649          movw    #cwpssrv$k_initial_maj_vers, cwpssrv$w_ext_maj_vers(R0)
1650          movw    #cwpssrv$k_initial_min_vers, cwpssrv$w_ext_min_vers(R0)
1651
1652          ; Stash the address of the post routine (not needed since there's
1653          ; nothing to clean up at this point).
1654
1655 ;        movab   CWPS$PCNTRL_CLEANUP,-           ; Move address of the routine
1656 ;                cwpssrv$a_post_routine(R0)      ;  to finish after the xfr
1657
1658          ; Call the common process control routine.  This routine will insert
1659          ; common data into the block, send the block to the remote node, and
1660          ; wait for return status.  After return, it will store the actual
1661          ; EPID of the affected process (if PIDADR<>0).  This routine assumes
1662          ; much of the current state, to wit:
1663          ;
1664          ;       PIDADR(AP) - Address of EPID for target process (optional)
1665          ;       PRCNAM(AP) - Address of process name   (optional)
1666          ;       R0         - Pointer to a service block, code and length set
1667          ;       R1         - EPID of remote process, or 0
1668          ;       R2         - CSID of remote node
1669          ;       R3         - scratch register
1670          ;       R4         - PCB address of current process
1671          ;       IPL        - IPL = IPL$_ASTDEL
1672
1673          jmp     g^CWPS$SSND_PCNTRL_RQST
1674
1675          ; The above routine executes the RET instruction which returns us to
1676          ; user code, this module does not regain control.
1677
1678 10$:     setipl  #0                              ; Restore original IPL
1679          brw     EXIT                            ; Exit if error
```

```
1681 ;       .sbttl  CWPS$PCNTRL_CLEANUP - Perform post-transfer activities
1682 ;+
1683 ;       Perform activities specific to common process control services which
1684 ;       must be done after the response from the remote node is available.
1685 ;
1686 ; Calling sequence:
1687 ;       JSB     @cwpssrv$a_post_routine(R6) from CWPS$SSND_PCNTRL_RQST
1688 ;
1689 ; Current state:
1690 ;       R0          - Status from remote node (CWPSSRV$L_STATUS(R6))
1691 ;       R1:R5       - undefined, available for scratch without save/restore
1692 ;       R6          - Pointer to the CWPSSRV$ service block
1693 ;       IPL         - IPL = 0
1694 ;
1695 ;       cwpssrv$l_status(R0)    - status from remote node
1696 ;
1697 ; Outputs from routine:
1698 ;       none.
1699 ;
1700 ;-
1701
1702 ;CWPS$PCNTRL_CLEANUP::
1703 ;
1704 ;       rsb
1705
1706         .END
```

6 SYSWAIT.LIS

```
SYSWAIT EVENT FLAG WAIT SERVICES 10-MAY-1989 15:51:40 VAX MACRO V5.0-8 Page 0
Table of contents

  (1)        96        DECLARATIONS
  (1)       123        EXE$WFLAND - WAIT FOR LOGICAL AND OF EVENT FLAGS
  (1)       168        EXE$WFLOR - WAIT FOR LOGICAL OR OF EVENTS
  (1)       213        EXE$WAITFR - WAIT FOR SINGLE EVENT
  (1)       260        EXE$WAIT - WAIT COMMON CODE
```

```
SYSWAIT EVENT FLAG WAIT SERVICES 10-MAY-1989 15:51:40 VAX MACRO V5.0-8 Page 1
X-7 28-APR-1988 10:58:38 [SYS.SRC]SYSWAIT.MAR;1 (1)

     1          .TITLE  SYSWAIT EVENT FLAG WAIT SERVICES
     2          .IDENT  'X-7'
     3
     4  ;*****************************************************************************
     5  ;*                                                                           *
     6  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
     7  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
     8  ;*  ALL RIGHTS RESERVED.                                                     *
     9  ;*                                                                           *
    10  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
    11  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
    12  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
    13  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
    14  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
    15  ;*  TRANSFERRED.                                                             *
    16  ;*                                                                           *
    17  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
    18  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
    19  ;*  CORPORATION.                                                             *
    20  ;*                                                                           *
    21  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
    22  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
    23  ;*                                                                           *
    24  ;*                                                                           *
    25  ;*****************************************************************************
    26  ;++
    27  ; FACILITY: EXECUTIVE, EVENT FLAG SERVICES
    28  ;
    29  ; ABSTRACT: WAIT CONTAINS THE THREE FORMS OF EVENT FLAG WAIT
    30  ;           SYSTEM SERVICES WHICH PROVIDE FOR SINGLE EVENT WAIT AS
    31  ;           WELL AS COMBINATIONS OF MULTIPLE EVENTS.
    32  ;
    33  ;--
    34  ;
    35  ; AUTHOR:
    36  ;       R.HUSTVEDT : VERSION
    37  ;
    38  ; MODIFIED BY:
    39  ;
    40  ;       X-7     SJF             Stu Farnham             28-Apr-1988
    41  ;               Synchronize EXE$WAIT with kernel ASTs by raising IPL
    42  ;               to ASTDEL.
    43  ;
    44  ;       X-6     WCT0015         Ward C. Travis          6-Jan-1987
    45  ;               Update  outdated  SMPLOCK,  SMPUNLOCK uses  to  LOCK,
    46  ;               UNLOCK for SMP.
    47  ;
    48  ;       X-5     SSA0003         Stan Amway              22-Sep-1986
    49  ;               Move routine SCH$WAIT to RSE.MAR. The routine has
    50  ;               knowledge of both scheduling mechanisms and policy,
    51  ;               and therefore, should reside in the PROCESS_MANAGEMENT
    52  ;               loadable image.
    53  ;
    54  ;       X-2E3   HH0166          Hai Huang               07-Apr-1986
    55  ;               Resolve merge conflicts.
    56  ;
    57  ;       X-2     MSH0209         Michael S. Harvey       26-Nov-1985
```

```
58 ;                Tighten up the multiprocessing interlock mechanisms.
59 ;
60 ;        V03-007 SSA0002         Stan Amway              8-Mar-1984
61 ;                Subtract IOTA from automatic working set adjustment
62 ;                time reference in PHD on any entry to SCH$WAITx code.
63 ;                (Acknowledgements go to Wayne Cardoza and Larry Kenah,
64 ;                 who both collaborated on this change.)
65 ;
66 ;        V03-006 WMC0001         Wayne Cardoza           22-Feb-1984
67 ;                No reason to try to wake up swapper on every EFN wait.
68 ;
69 ;        V03-005 SSA0001         Stan Amway              5-Dec-1983
70 ;                Added support for outswap scheduling changes.
71 ;                Changed process wait code to store wait time in PCB
72 ;                as longword system absolute time.
73 ;
74 ;        V03-004 KDM0035         Kathleen D. Morse       14-Dec-1982
75 ;                Fix assembly switch for performance collection for
76 ;                kernel mode services executed on secondary processor.
77 ;
78 ;        V03-003 KDM0034         Kathleen D. Morse       13-Dec-1982
79 ;                Correct logic for secondary continuing execution of
80 ;                a process after a WAITCHK request is done by the primary.
81 ;
82 ;        V03-002 KDM0030         Kathleen D. Morse       18-Nov-1982
83 ;                Add IFPRIMARY logic that allows primary to execute
84 ;                secondary-specific code without turning into a secondary.
85 ;
86 ;        V03-001 KDM0018         Kathleen D. Morse       13-Oct-1982
87 ;                Add multi-processing switch, which generates these
88 ;                system services for the secondary processor.
89 ;
90 ;        V04-001 SF04001         Stephen Fiorelli        27-Oct-1985
91 ;                System service macro used to declare entry point
92 ;                and build system service desriptor block.
93 ;                Added $SYSVECTORDEF.
94 ;
```

```
SYSWAIT EVENT FLAG WAIT SERVICES 10-MAY-1989 15:51:40 VAX MACRO V5.0-8 Page 3
X-7 DECLARATIONS 28-APR-1988 10:58:38 [SYS.SRC]SYSWAIT.MAR;1 (1)

   96          .SBTTL  DECLARATIONS
   97
   98 ;
   99 ; INCLUDE FILES:
  100 ;
  101
  102          $CEBDEF                                 ;COMMON EVENT BLOCK DEFS
  103          $DYNDEF                                 ;DYNAMIC STRUCTURE TYPES
  104          $EVTDEF                                 ;DEFINE SYSTEM EVENT CODES
  105          $IPLDEF                                 ;IPL DEFINITIONS
  106          $PCBDEF                                 ;PCB DEFINITIONS
  107          $PHDDEF                                 ;PHD DEFINITIONS
  108          $PRDEF                                  ;PROCESSOR REGISTER DEFS
  109          $PSLDEF                                 ;PSL DEFINITIONS
  110          $SHBDEF                                 ;SHARED MEMORY CONTROL BLK DEFS
  111          $SHDDEF                                 ;SHARED MEMORY COMMON DATA PAGE
  112          $SSDEF                                  ;STATUS CODE DEFINITIONS
  113          $STATEDEF                               ;STATE DEFINITIONS
  114          $SYSVECTORDEF                           ;SYSTEM SERVICE OFFSETS
  115          $WQHDEF                                 ;WAIT QUEUE HEADER DEFS
  116 ;
  117 ; EQUATES:
  118 ;
  119 EFN=4                                            ;EVENT FLAG NUMBER
  120 MASK=8                                           ;WAIT MASK
  121          DECLARE_PSECT   EXEC$NONPAGED_CODE      ;NON-PAGED
```

```
SYSWAIT EVENT FLAG WAIT SERVICES 10-MAY-1989 15:51:40 VAX MACRO V5.0-8 Page 4
X-7 EXE$WFLAND - WAIT FOR LOGICAL AND OF EVE 28-APR-1988 10:58:38 [SYS.SRC]SYSWAIT.MAR;1 (1)

  123          .SBTTL  EXE$WFLAND - WAIT FOR LOGICAL AND OF EVENT FLAGS
  124 ;++
  125 ; FUNCTIONAL DESCRIPTION:
  126 ;
  127 ;       EXE$WFLAND RETURNS TO THE CALLER WHEN THE SET OF EVENT
  128 ;       FLAGS SELECTED BY THE MASK ARE ALL SET AND RETURNS THE
  129 ;       STATE OF ALL EVENT FLAGS IN THE SPECIFIED CLUSTER.
  130 ;
  131 ; CALLING SEQUENCE:
  132 ;
  133 ;       CALLG   ARGLIST,EXE$WFLAND
  134 ;
  135 ; INPUT PARAMETERS:
  136 ;
  137 ;       04(AP) - EVENT FLAG NUMBER SELECTING CLUSTER
  138 ;       08(AP) - MASK SELECTING COMBINATION OF EVENTS
  139 ;       R4 - PCB ADDRESS OF CURRENT PROCESS
  140 ;
  141 ; OUTPUT PARAMETERS:
  142 ;
  143 ;       R0 - COMPLETION STATUS CODE
  144 ;            CONDITION IS SATISFIED.
  145 ;
  146 ; SIDE EFFECTS:
  147 ;
  148 ;       IF THE SET OF EVENT FLAGS SELECTED BY THE MASK ARE NOT
  149 ;       ALL SET, THEN THE PROCESS ISSUING THE WAIT SERVICE CALL WILL
  150 ;       BE PLACED IN A WAIT STATE.
  151 ;
  152 ; COMPLETION CODES:
  153 ;
  154 ;       SS$_NORMAL - NORMAL SUCCESSFUL COMPLETION
  155 ;       SS$_ILLEFC - ILLEGAL EVENT FLAG CLUSTER NUMBER.  EVENT NUMBER
  156 ;                    NOT IN THE RANGE 0-127.
  157 ;       SS$_UNASEFC - UNASSIGNED EVENT FLAG CLUSTER.
  158 ;
  159 ;--
  160
  161          SYSTEM_SERVICE  WFLAND,-
  162                          <R2,R3,R4,R5,R6>,-
  163                          MODE=KERNEL,-
  164                          NARG=2
  165          MOVL    #1,R1                   ;SET MODE TO WAITALL
  166          BRB     WFRL                    ;AND MERGE WITH COMMON CODE
```

```
SYSWAIT EVENT FLAG WAIT SERVICES 10-MAY-1989 15:51:40 VAX MACRO V5.0-8 Page 5
X-7 EXE$WFLOR - WAIT FOR LOGICAL OR OF EVENT 28-APR-1988 10:58:38 [SYS.SRC]SYSWAIT.MAR;1 (1)

  168          .SBTTL  EXE$WFLOR - WAIT FOR LOGICAL OR OF EVENTS
  169 ;++
  170 ; FUNCTIONAL DESCRIPTION:
  171 ;
  172 ;       EXE$WFLOR RETURNS TO THE CALLER WHEN ANY OF THE
  173 ;       EVENTS SELECTED BY THE MASK WITHIN THE SPECIFIED CLUSTER
  174 ;       ARE SET AND RETURNS THE STATE OF ALL 32 EVENT FLAGS IN THE
  175 ;       CLUSTER.
  176 ;
  177 ; CALLING SEQUENCE:
  178 ;
  179 ;       CALLG   ARGLIST,EXE$WFLOR
  180 ;
  181 ; INPUT PARAMETERS:
  182 ;
  183 ;       04(AP) - EVENT FLAG NUMBER TO SELECT CLUSTER
  184 ;       08(AP) - MASK SELECTING DESIRED COMBINATION OF EVENTS
  185 ;       R4 - PCB ADDRESS OF CURRENT PROCESS
  186 ;
  187 ; OUTPUT PARAMETERS:
  188 ;
  189 ;       R0 - COMPLETION STATUS CODE
  190 ;            IS SATISFIED.
  191 ;
  192 ; COMPLETION CODES:
  193 ;
  194 ;       SS$_NORMAL - NORMAL SUCCESSFUL COMPLETION
  195 ;       SS$_ILLEFC - ILLEGAL EVENT FLAG NUMBER NOT IN THE RANGE 0-127.
  196 ;       SS$_UNASEFC - UNASSIGNED EVENT FLAG CLUSTER.
  197 ;
  198 ; SIDE EFFECTS:
  199 ;
  200 ;       THE PROCESS ISSUING THE SERVICE CALL IS BE PLACED IN A
  201 ;       WAIT STATE IF NONE OF THE SPECIFIED EVENTS ARE SET.
  202 ;
  203 ;--
  204
  205          SYSTEM_SERVICE  WFLOR,-
  206                          <R2,R3,R4,R5,R6>,-
  207                          MODE=KERNEL,-
  208                          NARG=2
  209          CLRL    R1                      ;SET MODE TO WAIT ANY
  210 WFRL:    MOVL    MASK(AP),R0             ;GET WAIT MASK
  211          BRB     EXE$WAIT                ;MERGE WITH COMMON CODE
```

```
SYSWAIT EVENT FLAG WAIT SERVICES 10-MAY-1989 15:51:40 VAX MACRO V5.0-8 Page 6
X-7 EXE$WAITFR - WAIT FOR SINGLE EVENT 28-APR-1988 10:58:38 [SYS.SRC]SYSWAIT.MAR;1 (1)

  213          .SBTTL  EXE$WAITFR - WAIT FOR SINGLE EVENT
  214 ;++
  215 ; FUNCTIONAL DESCRIPTION:
  216 ;
  217 ;       EXE$WAITFR RETURNS TO THE CALLER WHEN THE SPECIFIED SINGLE
  218 ;       EVENT FLAG IS SET.  UPON RETURN THE STATE OF ALL 32 EVENT FLAGS
  219 ;       WITHIN THE CLUSTER CONTAINING THE SPECIFIED EVENT ARE RETURN.
  220 ;
  221 ; CALLING SEQUENCE:
  222 ;
  223 ;       CALLG   ARGLIST,EXE$WAITFR
  224 ;
  225 ; INPUT PARAMETERS:
  226 ;
  227 ;       04(AP)=EVENT FLAG NUMBER
  228 ;       R4 - PCB ADDRESS OF CURRENT PROCESS
  229 ;
  230 ; OUTPUT PARAMETERS:
  231 ;
  232 ;       R0 - COMPLETION STATUS CODE
  233 ;            SATISIFIED.
  234 ;
  235 ; SIDE EFFECTS:
  236 ;
  237 ;       IF THE SPECIFIED EVENT FLAG IS NOT SET, THE PROCESS ISSUING THE
  238 ;       WAIT SYSTEM SERVICE WILL BE PLACED IN THE APPROPRIATE WAIT
  239 ;       STATE.
  240 ;
  241 ; COMPLETION CODES:
  242 ;
  243 ;       SS$_NORMAL - NORMAL SUCCESSFUL COMPLETION
  244 ;       SS$_ILLEFC - ILLEGAL EVENT FLAG NUMBER NOT IN THE RANGE 0-127.
  245 ;       SS$_UNASEFC - UNASSIGNED EVENT FLAG CLUSTER.
  246 ;
  247 ; ENVIRONMENT:
  248 ;
  249 ;       MODE=KERNEL
  250 ;--
  251
  252          SYSTEM_SERVICE  WAITFR,-
  253                          <R2,R3,R4,R5,R6>,-
  254                          MODE=KERNEL,-
  255                          NARG=1
  256          CLRL    R1                      ;SET MODE
  257          ROTL    EFN(AP),#1,R0           ;INIT MASK
  258 ;        BRB     EXE$WAIT                ;AND MERGE WITH COMMON CODE
```

```
                        .SBTTL  EXE$WAIT - WAIT COMMON CODE
                ;++
                ; FUNCTIONAL DESCRIPTION:
                ;
                ;       THIS IS THE COMMON WAIT CODE FOR ALL THE EVENT FLAG WAIT
                ;       SYSTEM SERVICES.
                ;
                ; INPUT PARAMETERS:
                ;
                ;       04(AP) = EVENT FLAG NUMBER
                ;       R0 = MASK SELECTING EVENTS OF INTEREST
                ;       R1 = ANY/ALL MODE SELECTOR
                ;               0 => ANY
                ;               1 => ALL
                ;       R4 = PCB ADDRESS OF CURRENT PROCESS
                ;
                ; IMPLICIT INPUTS:
                ;
                ;       CEB IF NON-LOCAL CLUSTER.
                ;       SCHED DATABASE IS NOT LOCKED
                ;
                ; OUTPUT PARAMETERS:
                ;
                ;       R0 - COMPLETION STATUS CODE
                ;            SATISIFIED.
                ;
                ;--
                
EXE$WAIT:                                       ;WAIT COMMON CODE
                CVTBL   EFN(AP),R2              ;GET CLUSTER NUMBER
                BLSS    10$                     ;ILLEGAL IF NOT (0,1,2,3)
                ASHL    #-5,R2,R2               ;RIGHT ALIGN CLUSTER NUMBER
                SETIPL  #IPL$_ASTDEL            ;SYNCH WITH KERNEL AST LEVEL
                MOVAL   PCB$L_EFCS(R4)[R2],R3   ;POINTER TO PCB EVENT CLUSTER
                MOVB    R2,PCB$B_WEFC(R4)       ;SAVE WAIT CLUSTER NUMBER
                SOBGTR  R2,30$                  ;BR IF COMMON CLUSTER R2 = (2,3)
                MOVAQ   G^SCH$GQ_LEFWQ,R2       ;SET WAIT QUEUE POINTER
                BRB     WAITCK                  ;
10$:            MOVZWL  #SS$_ILLEFC,R0          ;SET ERROR CODE FOR ILLEGAL CLUSTER
                RET                             ;AND EXIT
20$:            MOVZWL  #SS$_UNASEFC,R0         ;SET ERROR CODE FOR UNASSIGNED
                RET                             ;AND EXIT
30$:            ADDL3   #CEB$L_EFC,(R3),R2      ;GET CEB ADDRESS FOR EVENT FLAGS
                BGEQ    20$                     ;CEB ASSIGNED (SYSTEM SPACE ADDRESS)
                CMPB    #DYN$C_SLAVCEB,<CEB$B_TYPE-CEB$L_EFC>(R2) ;IS THIS IN SH MEM?
                BNEQ    40$                     ;BR IF IN LOCAL MEMORY
                MOVL    <CEB$L_MASTER-CEB$L_EFC>(R2),R3 ;GET ADR OF MASTER CEB
                MOVL    CEB$L_EFC(R3),(R2)      ;COPY EFC FROM MASTER TO SLAVE CEB
                ASSUME  <CEB$L_EFC+4> EQ CEB$L_WQFL
40$:            MOVAL   (R2)+,R3                ;GET EVENT POINTER AND WAIT QUEUE ADDR
                                                ; R3=CEB$L_EFC, R2=CEB$L_WQFL
                ;
                ;       R0 - MASK SELECTING EVENTS OF INTEREST
                ;       R1 - ANY/ALL MODE SELECTOR
                ;       R2 - ADDRESS OF WAIT QUEUE HEADER
                ;       R3 - ADDRESS OF EVENT FLAG VECTOR
                ;       R4 - PCB ADDRESS
```

```
SYSWAIT EVENT FLAG WAIT SERVICES 10-MAY-1989 15:51:40 VAX MACRO V5.0-8 Page 8
X-7 EXE$WAIT - WAIT COMMON CODE 28-APR-1988 10:58:38 [SYS.SRC]SYSWAIT.MAR;1 (1)

  317 ;
  318 WAITCK:                                     ;CHECK FOR WAIT SATISFIED
  319         LOCK    LOCKNAME=SCHED,-            ;LOCK SCHED DATABASE - STOP SCHEDULING
  320                 LOCKIPL=#IPL$_SYNCH         ;RAISE IPL
  321         BITL    (R3),R0                     ;WAIT FOR LOGICAL OR MAY BE SATISFIED
  322         BEQL    WAIT                        ;NO, MUST WAIT
  323         BLBS    R1,WAITALL                  ; 1 => WAIT FOR ALL IN MASK
  324 NOWAIT:                                     ;
  325         MOVL    12(FP),FP                   ;GET SAVED FRAME POINTER
  326         ADDL    S^#EXE$C_CMSTKSZ,SP         ;CLEAN STACK TO PC,PSL
  327         MOVZBL  #SS$_NORMAL,R0              ;RETURN SUCCESS CODE
  328         UNLOCK  LOCKNAME=SCHED              ;UNLOCK SCHED DATABASE
  329         REIMAC                              ;RETURN TO CALLER
  330
  331 WAITALL:                                    ; WAIT FOR ALL SELECTED EVENTS
  332         BICL2   (R3),R0                     ; CLEAR BITS FOR ALREADY SET FLAGS
  333         BEQL    NOWAIT                      ;YES, DONT WAIT
  334 WAIT:   INSV    R1,#PCB$V_WALL,#1,PCB$L_STS(R4) ;SET WAIT ALL FLAG
  335         MCOML   R0,PCB$L_EFWM(R4)           ;SAVE INVERTED WAIT MASK
  336         JMP     G^SCH$WAIT
  337
  338         .END
```

7 EXEC_LAYOUT.LIS

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 0
Table of contents

  (4)       240         LOCAL SYMBOL DEFINITIONS
  (5)       247         SYSTEM SERVICE VECTOR AREA
  (6)       277         EXECUTIVE ROUTINE VECTOR AREA
  (7)       306         NONPAGED DATA AREA
  (8)       318         PAGED VECTORS AND DATA
  (9)       343         SYSPARAM DATA AREA
  (10)      355         BOOPARAM DATA AREA
  (10)      369         PATCH AREA
```

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 1
X-9 27-APR-1989 17:18:42 [SYS.SRC]EXEC_LAYOUT.MAR;1 (1)

    1          .TITLE  EXEC_LAYOUT
    2          .IDENT  'X-9'
    3 ;
    4 ;*****************************************************************************
    5 ;*                                                                           *
    6 ;*  COPYRIGHT (c) 1985 BY                                                    *
    7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
    8 ;*  ALL RIGHTS RESERVED.                                                     *
    9 ;*                                                                           *
   10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
   11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
   12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
   13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
   14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
   15 ;*  TRANSFERRED.                                                             *
   16 ;*                                                                           *
   17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
   18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
   19 ;*  CORPORATION.                                                             *
   20 ;*                                                                           *
   21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
   22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
   23 ;*                                                                           *
   24 ;*                                                                           *
   25 ;*****************************************************************************
   26 ;++
   27 ; Facility:
   28 ;
   29 ;       Executive Transfer Vector (Base image)
   30 ;
   31 ; Abstract:
   32 ;
   33 ; Define the layout of the VMS executive.
   34 ;
   35 ;
   36 ; Note:
   37 ;
   38 ;
   39 ;
   40 ; Author:
   41 ;
   42 ;       Stephen Fiorelli
   43 ;
   44 ; Creation Date:
   45 ;
   46 ;       6 May 1985
   47 ;
   48 ; Modified by:
   49 ;
   50 ;       X-9     GN00001         Gazo Namoglu            12-Apr-1989
   51 ;               Reserve space as a patch area referenced within
   52 ;               MMG$A_EXEC_PATCH_BEGIN and MMG$A_EXEC_PATCH_END.
   53 ;
   54 ;       X-8     SF00003         Stephen Fiorelli        16-Apr-1987
   55 ;               Plac end of base image holder here.
   56 ;
   57 ;       X-7     WMC0001         Wayne Cardoza           21-Apr-1986
```

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 2
X-9 27-APR-1989 17:18:42 [SYS.SRC]EXEC_LAYOUT.MAR;1 (1)

   58 ;               New mechanism for defining global addresses.
   59 ;
   60 ;       X-6     SF04002         Stephen Fiorelli        03-Feb-1986
   61 ;               Change lexical processing in the local $GBLINI macro
   62 ;               to strip off the BASE$ produced from exe_offsets.mar
   63 ;               and sys_offsets.mar
   64 ;
   65 ;       X-5     SF04001         Stephen Fiorelli        10-Jan-1986
   66 ;               Change the local $GBLINI macro to generate SYS$name
   67 ;               symbols or (prefix)$name symbols depending on a switch.
   68 ;
   69 ;       X-4     TCM0003         Trudy C. Matthews       26-Nov-1985
   70 ;               In re-definition of $GBLINI, generate EXE$name symbols
   71 ;               for system service vectors rather than SYS$name symbols.
   72 ;
   73 ;       V04-002 TCM0002         Trudy C. Matthews       3-Oct-1985
   74 ;               Correct bug in $GBLINI macro - make sure we store the
   75 ;               JMP/NOP/NOP sequence at the pre-determined vector offset.
   76 ;
   77 ;       V04-001 TCM0001         Trudy C. Matthews       29-Sep-1985
   78 ;               Add $MTACCESS routine offset.
   79 ;--
```

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 3
X-9 27-APR-1989 17:18:42 [SYS.SRC]EXEC_LAYOUT.MAR;1 (3)

   81
   82
   83          .DISABLE TRACEBACK
   84
   85 ;+
   86 ; $GBLINI - INITIALIZE THE GLOBAL/LOCAL DEFINITION SWITCH
   87 ;       This is a special local definition of $GBLINI.
   88 ;       If BASE_ADDRESS is defined when this macro is invoked,...
   89 ;       Also, if USE_SYS_PREFIX = 1 when this macro is invoked, the string
   90 ;       'SYS$' will be used instead of the prefix defined in the symbol
   91 ;       name.
   92 ;
   93 ;       GBL   = GLOBAL, LOCAL, OR NULL.
   94 ;               IF THIS PARAMETER IS GLOBAL
   95 ;               GLOBAL DEFINITIONS ARE GENERATED.  OTHERWISE
   96 ;               LOCAL DEFINITIONS ARE GENERATED.
   97 ;
   98 ;       This local $GBLINI macro was created to define the
   99 ;       symbols associated with exec vectors.
  100 ;       THE  macro definitions are affected by this $GBLINI;
  101 ;       is $SYSVECTORDEF.  $SYSVECTORDEF will be redefined
  102 ;       to produce symbols for system service vectors.
  103 ;       $SYSVECTORDEF was produced from the file
  104 ;       SYS_OFFSETS.SDL.  Definitions within this macro
  105 ;       are of the form:
  106 ;
  107 ;       $EQU    BASE$EXE$K_QIO  456
  108 ;
  109 ;       Since $DEFINI (defines the macros $SYSVECTORDEF
  110 ;       produced by the SDL complier)
  111 ;       uses $GBLINI, this local $GBLINI
  112 ;       will produce EXE$QIO (from BASE$EXE$K_QIO) equal
  113 ;       to a BASE_ADDRESS (if defined) plus the offset
  114 ;       from the base.
  115 ;
  116 ;       $EQU    BASE$EXE$K_QIO  456
  117 ;
  118 ;       would produce
  119 ;
  120 ;       EXE$QIO == 800001C8
  121 ;
  122 ;       in this module with a base address defined to be
  123 ;       80000000.
  124 ;-
  125         .MACRO  $GBLINI GBL=LOCAL
  126         .IF     IDN <GBL> <GLOBAL>
  127         .MACRO  $DEF    SYM,ALLOC,SIZ
  128         .IIF    NB,SYM, SYM::
  129         .IIF    NB,ALLOC,       ALLOC   SIZ
  130         .ENDM   $DEF
  131         .IF     NOT_DEFINED     BASE_ADDRESS
  132                 .MACRO  $EQU    SYM,VAL
  133                 SYM==VAL
  134                 .ENDM   $EQU
  135         .IF_FALSE
  136                 .MACRO  $EQU    NAME , VALUE
  137                 ...LEN... = %LENGTH(NAME)
```

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 4
X-9 27-APR-1989 17:18:42 [SYS.SRC]EXEC_LAYOUT.MAR;1 (3)

                ...BASE_LEN... = %LOCATE(<$>,NAME)+1
                ...PREFIX_LEN... = %LOCATE(<K_>,NAME)-...BASE_LEN...
                .IF     EQUAL   <...LEN...-...PREFIX_LEN...>
                        .ERROR  ; String name does not contain K_
                .IF_FALSE
                        ...SUFFIX_LEN... = ...LEN...-<%LOCATE(<K_>,NAME)+2>
                        ...SUFFIX... = %LOCATE(<K_>,NAME)+2

                        .IF     EQ,USE_SYS_PREFIX

        %EXTRACT(...BASE_LEN...,...PREFIX_LEN...,NAME)%EXTRACT(...SUFFIX...,...SUFFI
                                ==  BASE_ADDRESS + VALUE

                        .IF_FALSE

        SYS$%EXTRACT(...SUFFIX...,...SUFFIX_LEN...,NAME)-
                                 == BASE_ADDRESS + VALUE

                        .ENDC

                        .RESTORE_PSECT
                        . = VALUE
                        JMP     @#EXE$LOAD_ERROR
                        NOP
                        NOP
                        .SAVE_PSECT
                        .PSECT  $ABS$,ABS

                .ENDC
                .ENDM   $EQU
        .ENDC
        .MACRO  $VIELD1 MOD,SEP,SYM,SIZ,MSK
        SIZ...=1
        .IIF    NB,SIZ, SIZ...=SIZ
        .IF     NB,SYM
        MOD'SEP'V_'SYM==BIT...
        .IIF    NB,SIZ, MOD'SEP'S_'SYM==SIZ
        .IIF    NB,MSK, MOD'SEP'M_'SYM==<<<1@SIZ...>-1>@BIT...>
        .ENDC
        BIT...=BIT...+SIZ...
        .ENDM   $VIELD1
        .IFF
        .IIF    DIF <GBL> <LOCAL>, -
                .ERROR ;ARG MUST BE GLOBAL,LOCAL,OR NULL;
        .MACRO  $DEF    SYM,ALLOC,SIZ
        .IIF    NB,SYM, SYM:
        .IIF    NB,ALLOC,       ALLOC   SIZ
        .ENDM   $DEF
        .IF     NOT_DEFINED     BASE_ADDRESS
                .MACRO  $EQU    SYM,VAL
                SYM=VAL
                .ENDM   $EQU

        .IF_FALSE

                .MACRO  $EQU    NAME , VALUE
                ...LEN... = %LENGTH(NAME)
```

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 5
X-9 27-APR-1989 17:18:42 [SYS.SRC]EXEC_LAYOUT.MAR;1 (3)

                ...BASE_LEN... = %LOCATE(<$>,NAME)
                ...PREFIX_LEN... = %LOCATE(<K_>,NAME)
                .IF     EQUAL   <...LEN...-...PREFIX_LEN...>
                        .ERROR  ; String name does not contain K_
                .IF_FALSE
                        ...SUFFIX_LEN... = ...LEN...-<%LOCATE(<K_>,NAME)+2>
                        ...SUFFIX... = %LOCATE(<K_>,NAME)+2

                        .IF     EQ,USE_SYS_PREFIX


        %EXTRACT(...BASE_LEN...,...PREFIX_LEN...,NAME)%EXTRACT(...SUFFIX...,...SUFFI
                                ==  BASE_ADDRESS + VALUE

                        .IF_FALSE

        SYS$%EXTRACT(...SUFFIX...,...SUFFIX_LEN...,NAME)-
                                 = BASE_ADDRESS + VALUE

                        .ENDC

                        .RESTORE_PSECT
                        . = VALUE
                        NOP
                        NOP
                        JMP     @#EXE$LOAD_ERROR
                        .SAVE_PSECT
                        .PSECT  $ABS$,ABS

                .ENDC
                .ENDM   $EQU
        .ENDC
        .MACRO  $VIELD1 MOD,SEP,SYM,SIZ,MSK
        SIZ...=1
        .IIF    NB,SIZ, SIZ...=SIZ
        .IF     NB,SYM
        MOD'SEP'V_'SYM=BIT...
        .IIF    NB,SIZ, MOD'SEP'S_'SYM=SIZ
        .IIF    NB,MSK, MOD'SEP'M_'SYM=<<<1@SIZ...>-1>@BIT...>
        .ENDC
        BIT...=BIT...+SIZ...
        .ENDM   $VIELD1
        .ENDC
        .ENDM   $GBLINI

        .SBTTL  LOCAL SYMBOL DEFINITIONS
SYS$C_S0_VECTOR_PAGES == 16             ; # pages in system service vector area
EXE$C_VECTOR_PAGES == 16                ; # pages in exec routine vector area
EXE$C_NPAG_DATA_PAGES == 32             ; # pages in non-paged data area
EXE$C_SYSPARAM_PAGES == 16              ; # pages in system parameter area
EXE$C_BOOPARAM_PAGES == 4               ; # pages in boot parameter area
```

```
  247         .SBTTL  SYSTEM SERVICE VECTOR AREA
  248 ;++
  249 ; SYSTEM SERVICE VECTOR AREA
  250 ;
  251 ;       Reserve space in the base system image for all system service vectors.
  252 ;       Also, define a global symbol for each vector location.
  253 ;--
  254         .PSECT  $$$$$000_SYSTEM_SERVICE_VECTORS,PAGE,NOEXE
  255
  256 SYS$S0_VECTOR_BASE::                    ; Define base of system service vector
  257                                         ; area.
  258
  259 BASE_ADDRESS = SYS$S0_VECTOR_BASE       ;Define flag for $GBLINI
  260 USE_SYS_PREFIX = 0                      ; All other globals are defined
  261                                         ; using prefixes in the symbol names.
  262
  263
  264         $SYSVECTORDEF   GLOBAL          ;Define global symbols for system
  265                                         ;service transfer vectors in this region
  266
  267 SYS$S0_VECTOR_LAST_USED::              ; Define space actually used by
  268                                         ; vectors.
  269         .BLKB   <512*SYS$C_S0_VECTOR_PAGES> - <.-SYS$S0_VECTOR_BASE>
  270                                         ; Reserve rest of pages for system
  271                                         ; service vectors.
  272
  273 SYS$S0_VECTOR_END::                     ; Define end of system service vector
  274                                         ; area.
  275
```

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 7
X-9 EXECUTIVE ROUTINE VECTOR AREA 27-APR-1989 17:18:42 [SYS.SRC]EXEC_LAYOUT.MAR;1 (6)

  277          .SBTTL  EXECUTIVE ROUTINE VECTOR AREA
  278 ;++
  279 ; NON-PAGED EXECUTIVE ROUTINE VECTOR AREA
  280 ;--
  281          .PSECT  $$$$$NONPAGED_CODE,PAGE,EXE,OVR
  282
  283 EXE$VECTOR_BASE::                          ; Define base of exec subroutine vector
  284                                             ; area.
  285
  286 BASE_ADDRESS = EXE$VECTOR_BASE              ;Define flag for $GBLINI
  287
  288
  289          .BLKB   <512*EXE$C_VECTOR_PAGES> - <.-EXE$VECTOR_BASE>
  290                                             ; Reserve space for exec subroutine
  291                                             ; vectors.
  292
  293 EXE$VECTOR_END::                            ; Define end of executive subroutine
  294                                             ; vector area.
  295
  296 ;
  297 ; Note special fix-ups needed for routines that can be contained within
  298 ; the 8-byte vector space:
  299 ;        EXE$LOAD_ERROR: HALT
  300 ;        EXE$LOAD_NOP: RSB
  301 ;        IOC$RETURN: RSB
  302 ;        INI$BRK: NOP followed by RSB
  303 ;
  304
```

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 8
X-9 NONPAGED DATA AREA 27-APR-1989 17:18:42 [SYS.SRC]EXEC_LAYOUT.MAR;1 (7)

  306          .SBTTL  NONPAGED DATA AREA
  307 ;++
  308 ; NON-PAGED EXECUTIVE DATA AREA
  309 ;--
  310          .PSECT  $$$$$NONPAGED_DATA,PAGE,WRT,NOEXE,OVR
  311 MMG$A_NPAG_DATA::                       ; Define start of non-paged data area.
  312
  313          .BLKB   <512*EXE$C_NPAG_DATA_PAGES>
  314                                          ; Reserve space for non-paged data.
  315
  316 MMG$A_NPAG_DATA_END::                   ; Define end of non-paged data area.
```

```
                        .SBTTL  PAGED VECTORS AND DATA
;*****************************************************************************;
; ***NOTE*** For now, the base image does not contain any paged vector or     ;
; data areas.  If paged code/data is to be added to the base image, the       ;
; code in SYSBOOT that loads the base image into memory will have to change   ;
; to take this into account.                                                  ;
;*****************************************************************************;

;++
; PAGED EXECUTIVE ROUTINE VECTOR AREA
;--
                        .PSECT  $$$$$PAGED_CODE,PAGE,EXE
; This is a null placeholder for now.
MMG$A_PAGED_CODE::
MMG$A_PAGED_CODE_END::

;++
; PAGED EXECUTIVE DATA AREA
;--
                        .PSECT  $$$$$PAGED_DATA,PAGE,WRT,NOEXE,OVR
; This is a null placeholder for now.
MMG$A_PAGED_DATA::
MMG$A_PAGED_DATA_END::

```

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 10
X-9 SYSPARAM DATA AREA 27-APR-1989 17:18:42 [SYS.SRC]EXEC_LAYOUT.MAR;1 (9)

  343          .SBTTL  SYSPARAM DATA AREA
  344 ;++
  345 ; SYSPARAM DATA AREA
  346 ;--
  347          .PSECT  $$$$$SYSPARAM_DATA,PAGE,WRT,NOEXE,OVR
  348 MMG$A_SYSPARAM::                        ; Define start of SYSPARAM area.
  349
  350          .BLKB   <512*EXE$C_SYSPARAM_PAGES>
  351                                          ; Reserve space for SYSGEN parameters.
  352
  353 MMG$A_SYSPARAM_END::                     ; Define end of SYSPARAM area.
```

```
EXEC_LAYOUT 10-MAY-1989 14:57:55 VAX MACRO V5.0-8 Page 11
X-9 BOOPARAM DATA AREA 27-APR-1989 17:18:42 [SYS.SRC]EXEC_LAYOUT.MAR;1 (10)

355          .SBTTL  BOOPARAM DATA AREA
356 ;++
357 ; BOOPARAM DATA AREA
358 ;--
359          .PSECT  $$$$$Z_BOOPARAM_DATA,PAGE,WRT,NOEXE,OVR
360 MMG$A_BOOPARAM::                          ; Define start of BOOPARAM area.
361
362          .BLKB   <512*EXE$C_BOOPARAM_PAGES>
363                                           ; Reserve space for BOOPARAM parameters.
364
365 MMG$A_BOOPARAM_END::                       ; Define end of BOOPARAM area.
366
367 REORGANIZED_EXEC_END::                            ; Define end of reorganized exec are
368
369          .SBTTL  PATCH AREA
370 ;++
371 ; EXEC PATCH AREA
372 ;--
373          .PSECT  _Z_PATCH_AREA
374
375 MMG$A_EXEC_PATCH_BEGIN::                          ; Define the start of patch area.
376
377          .BLKB   <512/4>
378
379 MMG$A_EXEC_PATCH_END::                            ; Define the end of patch area.
380
381          .PSECT  _Z_SYS$END
382
383 MMG$A_SYS_END::
384          .LONG   0
385
386          .END
```

8 SYSPARAM.LIS

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 0
Table of contents
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 1
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

     1 ;*****************************************************************************
     2 ;*                                                                           *
     3 ;*                                NOTICE !!!                                 *
     4 ;*                                                                           *
     5 ;*  $SYIDEF IN MASD$:<VMSLIB.SRC>STARDEFQZ.SDL MUST BE UPDATED WHEN ADDING   *
     6 ;*  PARAMETERS - BUT NOT WHEN DELETING PARAMETERS. $SYIDEF CONTAINS THE      *
     7 ;*  PUBLIC DEFINITIONS OF THE $GETSYI ITEM-CODES, AND EACH SYSGEN PARAMETER  *
     8 ;*  IS AUTOMATICALLY TRANSFORMED INTO A $GETSYI ITEM.                        *
     9 ;*  SYITABLE.MAR DEFINES $GETSYI ITEMS WHICH ARE -NOT- SYSGEN PARAMETERS,    *
    10 ;*  AND SHOULD NOT BE MODIFIED WHEN ADDING OR SUBTRACTING SYSGEN PARAMETERS. *
    11 ;*                                                                           *
    12 ;*                          ** W A R N I N G **                              *
    13 ;*                                                                           *
    14 ;*      Sysgen parameter addresses cannot change in releases which do not    *
    15 ;*      require all privileged code to relink.  Thus, parameters cannot be   *
    16 ;*      removed without leaving placeholder cells, and new parameters must   *
    17 ;*      be added following the end of all previously defined parameters.     *
    18 ;*                                                                           *
    19 ;*****************************************************************************
    20
    21          .IF     DEFINED VERSION
    22          .TITLE  SYSPARAM_MASK - Define version masks for system parameters
    23          .IF_FALSE
    24          .IF     DEFINED GETSYISW
    25          .TITLE  GETSYI - DEFINE SYSTEM PARAMETERS FOR GETSYI
    26          .IF_FALSE
    27          .IF     NDF,PRMSW               ;
    28          .TITLE  SYSPARAM - SYSTEM PARAMETERS
    29          .IFF                            ;
    30          .TITLE  PARAMETER - PARAMETER DESCRIPTORS FOR SYSPARAM
    31          .ENDC
    32          .ENDC
    33          .ENDC
    34          .IDENT  'X-101U18'
    35
    36 ;
    37 ;*****************************************************************************
    38 ;*                                                                           *
    39 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984, 1988 BY                            *
    40 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
    41 ;*  ALL RIGHTS RESERVED.                                                     *
    42 ;*                                                                           *
    43 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
    44 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
    45 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
    46 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
    47 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
    48 ;*  TRANSFERRED.                                                             *
    49 ;*                                                                           *
    50 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
    51 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
    52 ;*  CORPORATION.                                                             *
    53 ;*                                                                           *
    54 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
    55 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
    56 ;*                                                                           *
    57 ;*                                                                           *
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 2
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

   58 ;*****************************************************************************
   59 ;
   60
   61 ;++
   62 ; FACILITY:  EXECUTIVE DATA BASE
   63 ;
   64 ; ABSTRACT:
   65 ;       SYSPARAM CONTAINS THE EXECUTIVE CONTROL PARAMETERS AND CERTAIN
   66 ;       KEY VARIABLES.
   67 ;
   68 ; ENVIRONMENT:
   69 ;
   70 ; AUTHOR:  R. I. HUSTVEDT, CREATION DATE:  09-OCT-1977
   71 ;
   72 ; MODIFIED BY:
   73 ;
   74 ;       X-101U18 RLF            Richard L. Frank        6-APR-1989
   75 ;               Change LCK$GL_IDTBLSIZ and LCK$GL_IDTBLMAX to support
   76 ;               262143 LKIDs.
   77 ;
   78 ;       X-101U17 RAB0008        Richard A. Bishop       31-Mar-1989
   79 ;               Change defaults for the following SYSGEN parameters
   80 ;               for the V5.2 SDC submission kit:
   81 ;               POOLCHECK       - from ^X6164001F to 0
   82 ;               BUGCHECKFATAL   - from          1 to 0
   83 ;               (i.e. reverse X101U12)
   84 ;
   85 ;       X-101U16 SJF            Stu Farnham             5-Jan-1989
   86 ;               Annual exercise in futility of updating base value
   87 ;               for EXE$GQ_TODCBASE. There must be a better way.
   88 ;
   89 ;       X-101U15 MAS0214        Mark A. Stiles          11-Nov-1988
   90 ;               Move tape alloclass name to the correct location.
   91 ;               Create some appropriate warnings about the ordering
   92 ;               of parameter addresses, and indicate where new
   93 ;               sysgen parameters should be defined.
   94 ;
   95 ;       X-101U14 CJB01011       Cheryl J. Bulmer        17-Oct-1988
   96 ;               Change tape alloclass name to TAPE_ALLOCLS.
   97 ;
   98 ;       X-101U13 WCT0128        Ward C. Travis          10-Oct-1988
   99 ;               Implement multi-page Error Log Buffers.
  100 ;
  101 ;       X-101U12 CWH51101U12    CW Hobbs                6-Oct-1988
  102 ;               Set POOLCHECK to ^x6164001F and BUGCHECKFATAL to 1
  103 ;               for field test.
  104 ;
  105 ;       X-101U11 CJB0101        Cheryl J. Bulmer        5-Oct-1988
  106 ;               Add tape allocation class cluster parameter,
  107 ;               CLU$GL_TAPE_ALLOCLS.
  108 ;
  109 ;       X-101U10 DOSU17         Dan O'Shaughnessy       5-Oct-1988
  110 ;               Change default value of SCSMAXMSG from 112 to 132.
  111 ;
  112 ;       X-101U9 TYC003          Theresa Chin            29-Sep-1988
  113 ;               Change default value of SCS$GB_PAMXPORT from 15 to 64.
  114 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 3
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  115 ;       X-101U8 WMC0U8          Wayne Cardoza           11-Sep-1988
  116 ;               Change default affinity_skip.
  117 ;
  118 ;       X-101U7 WMC0U7          Wayne Cardoza           11-Sep-1988
  119 ;               Lower maximum affinity_skip.
  120 ;
  121 ;       X-101U6 LPL0001         Lee Leahy               26 Aug 1988
  122 ;               REMACP has been rewritten and no longer has requirements
  123 ;               for strange limits on RJOBLIM.  All limits have been
  124 ;               removed for RJOBLIM.
  125 ;
  126 ;       X-101U5 SSA0028         Stan Amway              16-Aug-1988
  127 ;               Increase MPW_WRTCLUSTER default from 96 to 120.
  128 ;
  129 ;       X-101U4 SSA0027         Stan Amway              15-Jul-1988
  130 ;               Add SA_APP bit to EXE$GL_STATIC_FLAGS.
  131 ;
  132 ;       X-101U3 WMC00U3         Wayne Cardoza           30-Jun-1988
  133 ;               Add affinity control parameters.
  134 ;
  135 ;       X-101U2 SSA0026         Stan Amway              15-Jun-1988
  136 ;               Increase from 16384 to 65535 the maximum values for
  137 ;               MPW_HILIMIT, MPW_LOLIMIT, MPW_THRESH, MPW_WAITLIMIT,
  138 ;               and MPW_LOWAITLIMIT.
  139 ;
  140 ;       X-101U1 MWEDW1101U1     Mark Eaton              14-JUN-1988
  141 ;               V5.0 SDC changes didn't get folded back in
  142 ;
  143 ;       X-101   SLT007          Sarah Tappan            20-Apr-1988
  144 ;               same as 100, wrong file got replaced.
  145 ;
  146 ;       X-100   SLT0006         Sarah Tappan            19-Apr-1988
  147 ;               Add units field to mpw_thresh, mpw_waitlimit,mpw_lowaitlimit
  148 ;
  149 ;       X-99    MWE             Mark W. Eaton            7-APR-1988
  150 ;               Increase PQL_DPGFLQUOTA default to 8192
  151 ;
  152 ;       X-98U3  CBD0190         Charles B. Dunn         20-Apr-1988
  153 ;               Bump the maximum and default values of
  154 ;               PHYSICALPAGES to allow support of 511.5M of memory.
  155 ;
  156 ;       X-98U2  MMM0001         Marian M. Macartney     06-Apr-1988
  157 ;               Up PQL_DPGFLQUOTA default to 8192.
  158 ;
  159 ;       X-98U1  RAB0005         Richard A. Bishop       23-Mar-1988
  160 ;               Change defaults for the following SYSGEN parameters
  161 ;               for the SDC submission kit:
  162 ;               MULTIPROCESSING - from          1 to 3
  163 ;               POOLCHECK       - from ^X6164001F to 0
  164 ;               BUGCHECKFATAL   - from          1 to 0
  165 ;
  166 ;       X-98    SLT0005         Sarah Tappan            22-Mar-1988
  167 ;               Remove unused parameter DEFMBXNUMMSG
  168 ;
  169 ;       X-97    SLT0004         Sarah Tappan            1-Feb-1988
  170 ;               Modify window_systems parameter
  171 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 4
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  172 ;       X-96    HH0308          Hai Huang               13-Jan-1988
  173 ;               Update time-of-day-clock base to 1-Jan-1988.
  174 ;
  175 ;       X-95    GHC0035         George Claborn          31-Dec-1987
  176 ;               Change various defaults to allow (hopefully) all
  177 ;               configurations to boot.
  178 ;
  179 ;       X-94    SLT0003         Sarah Tappan            1-Dec-1987
  180 ;               Minor edit to PQL_MWSEXTENT parameter, change minimum to
  181 ;               match PQL_MWSQUOTA.
  182 ;
  183 ;       X-93    SSA0025         Stan Amway              17-Nov-1987
  184 ;               Add parameter RSRVPAGCNT to control the number of pages
  185 ;               that are reserved and escrowed for the current process page
  186 ;               file.
  187 ;
  188 ;               Change NOPGFLSWP from type SYS to SPECIAL so that only
  189 ;               SHOW/SPECIAL in SYSGEN will display it.
  190 ;
  191 ;       X-92    SF00092         Stephen Fiorelli        02-Nov-1987
  192 ;               Add version masks for sysgen
  193 ;
  194 ;       X-91    SJF             Stu Farnham             27-Oct-1987
  195 ;               Add mask forms of TIME_CONTROL bits
  196 ;
  197 ;       X-90    SLT0002         Sarah Tappan            22-Oct-1987
  198 ;               Make minor edit to tty-parity, add maximum value
  199 ;
  200 ;       X-89    MSH0323         Michael S. Harvey       5-Oct-1987
  201 ;               Put the BREAKPOINT mask into a different cell so it's
  202 ;               value can be used by INIT without INIT having to destroy it.
  203 ;
  204 ;       X-88    SSA0024         Stan Amway              25-Sep-1987
  205 ;               Move scheduling control flag bit definitions out of
  206 ;               this module, into RSE.MAR.
  207 ;
  208 ;               MJW0149         Michael J. Worcester    25-Sep-1987
  209 ;               Add the new SYSGEN parameter DUMPSTYLE to support
  210 ;               the new 512 Mb subset dump mechanism.
  211 ;
  212 ;       X-87    WES0190         William E. Snaman       11-Sep-1987
  213 ;               Change default for QDSKINTERVAL from 20 to 10 seconds.
  214 ;
  215 ;       X-86    MSH0323         Michael S. Harvey       11-Sep-1987
  216 ;               Add BREAKPOINTS parameter to give us control over the
  217 ;               set of initial breakpoint callers.
  218 ;
  219 ;               Also, set MAX value of MAXWSCNT from 64000 to 100000
  220 ;               for Steve Fiorelli.
  221 ;
  222 ;       X-85    WLG5085         William Goleman         11-Sep-1987
  223 ;               Add new parameters for controlling buffer size and
  224 ;               credits in the MSCP server. Change default for
  225 ;               SWPOUTPGCNT.
  226 ;
  227 ;       X-84    SSA0023         Stan Amway              8-Sep-1987
  228 ;               Add special SYSGEN parameter CLOCK_INTERVAL.
```

```
;
;       X-82    WLG5082         William Goleman         2-Sep-1987
;               Change the MSCP_SERVE_ALL parameter from a boolean
;               type parameter to counted string.
;
;       X-81    SLT0001         Srah Tappan             26-Aug-1987
;               remove obsolute parameter autoconfg_algo.  This
;               parameter was added as part of a sysgen bug fix
;               and never used.
;
;       X-80    PLL0000         Pam Levesque            21-Aug-1987
;               Add WINDOW_SYSTEM SYSGEN parameter for DECwindows.
;
;       X-79    JAY0001         John A. Ywoskus         18-Aug-1987
;               Rename LAVC_CONV_BOOT, LAVC_LOAD_PEA0, and LAVC_PORT_SERV
;               to NISCS_CONV_BOOT, NISCS_LOAD_PEA0, and NISCS_PORT_SERV.
;               Rename CLU$GQ_LAVC_AUTH and CLU$QL_LAVC_GROUP to
;               CLU$GQ_NISCS_AUTH and CLU$QL_NISCS_GROUP. Rename
;               CLU$V_LAVC_CONV_BOOT and CLU$V_LAVC_LOAD_PEA0 to
;               CLU$V_NISCS_CONV_BOOT and CLU$V_NISCS_LOAD_PEA0.
;               Change MVTIMEOUT default to 3600 and SCSRESPCNT to 300.
;
;       X-78    CWH5078         CW Hobbs                5-Aug-1987
;               Set default for BUGCHECKFATAL to on.
;
;       X-77    NCD0001         Nancy C. Denardo        30-Jul-1987
;               Add global symbol BOO$C_PRMBLKSZ which equates to
;               the total length of the parameter block descriptors
;               area.
;
;       X-76    SJF             Stu Farnham             28-Jul-1987
;               Make lengths of MINWSCNT , AWSMIN, and SYSDWSCNT agree with
;               code.
;       X-75    SF00075         Stephen Fiorelli        24-Jul-1987
;               Caught by sysgen parameter magic.  Eliminate _obs
;               symbols for minwscnt and sysdwscnt.
;
;       X-74    SF00074         Stephen Fiorelli        17-Jul-1987
;               Initial large working set support.  Add two new
;               cells sgn$gl_minwscnt, and sgn$gl_sysdwscnt to
;               replace the word versions.  The word versions will
;               remain for testing purposes, and will be removed
;               after the next baselevel.
;
;               Remove mmg$gl_pgdcod and its synonym pat$gl_exp_npg2
;               since they are obsolete.
;
;       X-73    CWH5073         CW Hobbs                20-Jun-1987
;               Set default for pool check to enabled for field
;               test.
;
;       X-72    RG00140         Raymond Guzman          15-May-1987
;               Added SYSGEN flag to support the job controller
;               upgrade and future diagnostics.
;
;       X-71    WCT0071         Ward C. Travis          16-Apr-1987
;               Make full multiprocessor operation the default
```

```
;                behavior  by setting  the default  for SMP_CPUS  to
;                all ones (-1).
;
;                WCT0069         Ward C. Travis          14-Apr-1987
;                Add parameter ERRORLOGBUFFERS, which  is hooked  up
;                to SGN$GW_ERLBUFCNT.
;
;        X-70    RLN026          Richard L. Napolitano   2-Apr-1987
;                Add AUTOCONFIG_ALGO parameter.
;
;        X-69    DDD             Dan D. Doherty          17-Mar-1987
;                Increased NPAGEDYN to 153600 bytes (300 pages) to ensure
;                successful loading of all images and structures during early
;                phases of boot when pool expansion cannot occur.
;
;        X-68    SSA0022         Stan Amway              17-Mar-1987
;                Change default for WSDEC to non-zero value to enable
;                WS reduction of computable, dormant processes.
;
;        X-67    WES0144         William E. Snaman       16-Mar-1987
;                Change default value of PRCPOLINTERVAL to 60 seconds.
;
;        X-66    SJF             Stu Farnham              9-Mar-1987
;                Change EXE$V_NO_SPINWAIT and NO_SANITY to NOSPINWAIT and
;                NOSANITY.
;
;        X-65    SSA0021         Stan Amway              3-Mar-1987
;                Fix problem with variant assembly of this module.
;
;        X-64    SSA0020         Stan Amway              2-Mar-1987
;         -63    Add MAXCLASSPRI, MINCLASSPRI and MINPRPRI.
;
;        X-62    SF04002         Stephen Fiorelli        26-Feb-1987
;                The cell exe$al_stacks is useless.  Kill it.
;                Also the parameters LOADERAPAT, LOADCHKPRT, and LOADMTACCESS
;                are no longer necessary with the new mechanism
;                for loading system images not part of the base system.
;
;        X-60/61 CWH5060         CW Hobbs                23-Feb-1987
;                Add:
;                 TAPE_MVTIMEOUT - tape mount verification timeout
;                 POOLCHECK      - controls for pool-checker in MEMORYALC
;                 PSEUDOLOA      - size of pseudo-image (tape boot flag)
;                 LAVC_CONV_BOOT - conversational boot allowed on remote boot
;                 LAVC_LOAD_PEA0 - load PEDRIVER, the NISCS module
;                 LAVC_PORT_SERV - port services flag for PEA0
;                 MSCP_LOAD      - load server at boot time
;                 MSCP_SERVE_ALL - serve all 'local' disks
;                Set MULTIPROCESSING max to 3 and default to 1.
;
;        X-58    LSS0027         Leonard S. Szubowicz    19-Feb-1987
;                1. Remove obsolete CJFLOAD and CJFSYSRUJ parameters.
;                2. Shorten SMP_LONGSPINWAIT to SMP_LNGSPINWAIT to meet
;                   PRM$C_MAXNAMLEN constraint.
;
;        X-57    SJF             Stu Farnham             10-Feb-1987
;                1. Break EXE$V_NOCLOCK out of EXE$GL_DEFFLAGS. The
```

```
343 ;                   NOCLOCK parameter is now obsolete, replaced by a set
344 ;                   of bits in the TIME_CONTROL paramter, with separate control
345 ;                   over hardware clock interrupts, SMP sanity timer,
346 ;                   and SMP spin/busy waits.
347 ;                2. Modify intervals for SMP_SPINWAIT and SMP_BUSYWAIT to
348 ;                   streamiline wait code.
349 ;                3. Modify untis for SMP_SANITY_CNT and SMP_TICK_CNT to
350 ;                   reflect modified semantics.
351 ;                4. Set default, maximum for PHYSICALPAGES to 532480 (change
352 ;                   made for Brian Porter)
353 ;                5. Add MULTIPROCESSING parameter type and include appropriate
354 ;                   SMP parameters in that type.
355 ;                6. Change SMP_LONGWAIT to SMP_LONGSPINWAIT
356 ;
357 ;       X-56    BP              Brian Porter,           27-JAN-1987
358 ;               Bump virtual page count.
359 ;
360 ;       X-55    SJF             Stu Farnham             27-Jan-1987
361 ;               TEMPORARILY increase the default values for SMP_SPINWAIT
362 ;               and SMP_LONGWAIT to avoid premature timeouts while
363 ;               experience is gained with the code. Scale parameters
364 ;               correctly.
365 ;
366 ;       X-54    WES             William E. Snaman       23-Jan-1987
367 ;               Increase the default value for LOCKIDTBL_MAX.
368 ;
369 ;       X-53    SJF             Stu Farnham             22-Jan-1987
370 ;               Set new upper bounds for SMP_SPINWAIT and SMP_LONGWAIT
371 ;               to prevent overflow on normalization.
372 ;
373 ;       X-52    LMP0430         L. Mark Pilant,         12-JAN-1987  12:19
374 ;               Up MAXBUF for changes made to DIRECTORY.
375 ;
376 ;       X-51    SSA0019         Stan Amway               7-Jan-1987
377 ;               Document code dependency on adjacency, and absolute or
378 ;               relative positions of bits in SCH$GL_CTLFLAGS. The
379 ;               code cannot use ASSUME statements.
380 ;
381 ;       X-50    HH0236          Hai Huang               05-Jan-1987
382 ;               Add a special parameter LOAD_SYS_IMAGES to control loading
383 ;               of system images described in the VMS$SYSTEM_IMAGES.EXE file.
384 ;
385 ;       X-49    SJF             Stu Farnham             22-Dec-1986
386 ;               Add SMP busywait timeout parameters SMP_SPINWAIT and
387 ;               SMP_LONGWAIT.
388 ;
389 ;       X-48    SSA0018         Stan Amway              12-Dec-1986
390 ;               Increase default value for MPW_IOLIMIT. Lower default
391 ;               value for DORMANTWAIT.
392 ;
393 ;       X-47    MAS0090         Mark A. Stiles           8-Dec-1986
394 ;               Add SHADOWING sysgen parameter for volume shadowing.
395 ;
396 ;       X-46    SJF             Stu Farnham             04-Dec-1986
397 ;               Shorten SMP_SANITY_COUNT and SMP_TICK_COUNT to SMP_
398 ;               SANITY_CNT and SMP_TICK_CNT
399 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 8
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  400 ;       X-44,45 JCK             Jonathan C. Kaplan      04-Dec-1986
  401 ;               add QBUS_MULT_INTR to control the initialization of
  402 ;               the multi-level interrupt dispatching code in the
  403 ;               CRB.
  404 ;
  405 ;       X-43    SJF             Stu Farnham             03-Dec-1986
  406 ;               Change names of SMP sanity timer parameters.
  407 ;
  408 ;       X-42    DSS0001         David Stonehill         02-DEC-1986
  409 ;               Change RJOBLIM minimum to 2, which is limit
  410 ;               enforced by code.
  411 ;
  412 ;       X-41    PAJ0017         Paul A. Jacobi          26-Nov-1986
  413 ;               Add WPRE_SIZE and WPTTE_SIZE parameter to control
  414 ;               allocation of nonpaged pool by the Watchpoint driver
  415 ;               (WPDRIVER).
  416 ;
  417 ;       X-40    SSA0017         Stan Amway              25-Nov-1986
  418 ;               Add CPU scheduling control flag, NO_QEND_PREEMPT.
  419 ;
  420 ;       X-39    SJF             Stu Farnham             20-Nov-1986
  421 ;               Add SMP timeout parameters.
  422 ;
  423 ;       X-38    RNH0052         Richard N. Holstein     17-Nov-1986
  424 ;               An artificial limit of 128MB for PHYSICALPAGES is too
  425 ;               restrictive for new VENUS and NAUTILUS memories; bump it up.
  426 ;
  427 ;       X-37    WES0100         William E. Snaman       14-Nov-1986
  428 ;               Change minimum value of EXPECTED_VOTES to 1.
  429 ;
  430 ;       X-36    WES0090         William E. Snaman       11-Nov-1986
  431 ;               Remove the FALLBACK_MODE parameter.
  432 ;
  433 ;       X-35    SSA0016         Stan Amway              11-Nov-1986
  434 ;               Add NOPGFLSWP, a dynamic SYSGEN flag.
  435 ;
  436 ;       X-34    WMC0034         Wayne Cardoza           5-Nov-1986
  437 ;               Kill the SYSPAGING parameter.
  438 ;
  439 ;       X-32    RLN006          Richard L. Napolitano   30-Oct-1986
  440 ;               1) ADD PUDRIVER_OPTIONS parameter.
  441 ;               2) Shorten name from PUDRIVER_OPTIONS to PU_OPTIONS.
  442 ;
  443 ;       X-31    SSA0015         Stan Amway              30-Oct-1986
  444 ;               Changed default for WSDEC to 0. In V4.x, since the default
  445 ;               for PFRATL is 0, the WSDEC default was immaterial.
  446 ;               In V5.x, new AWSA logic in RSE interprets PFRATL=0,
  447 ;               WSDEC <> 0 in a special way. Changing this default
  448 ;               preserves V4 behavior when PFRATL=0 (the common case).
  449 ;
  450 ;       X-29    SSA0014         Stan Amway              28-Oct-1986
  451 ;               Add CPU scheduling control flags, NO_CHSE_PRIADJ
  452 ;               and PIXSCAN_LOWPRI.
  453 ;
  454 ;       X-28    WES0077         William E. Snaman       24-Oct-1986
  455 ;               Remove the QUORUM parameter.
  456 ;
```

```
;       X-27    SSA0013         Stan Amway              14-Oct-1986
;               Make CPU scheduling control flags DYNAMIC and rearrange
;               flags so that BLBx can be used in SCHED.MAR for
;               a more optimal test.
;
;       X-26    WES0063         William E. Snaman       13-Oct-1986
;               Add expected votes parameter.  Lower recnxinterval
;               to 20 seconds.
;
;       X-25    SSA0012         Stan Amway              13-Oct-1986
;               Add a new CPU scheduling control flag.
;
;       X-24    RNG0024         Rod N. Gamache          9-Oct-1986
;               Add MULTIPROCESSING parameter.
;
;       X-23    SSA0011         Stan Amway              22-Sep-1986
;        -22    Add CPU scheduling control flags parameter.
;
;       X-21    WMC0015         Wayne Cardoza           15-Sep-1986
;               Units for SO_PAGING should be lower case.
;
;       X-20    WMC0014         Wayne Cardoza           05-Sep-1986
;               Add SO_PAGING parameter and its cell.
;
;       X-19    TCM             Trudy C Matthews        02-Sep-1986
;               Add EXE$GW_CPUMODEL cell.
;
;       X-18    SSA0010         Stan Amway              28-Aug-1986
;               Add MPW_IOLIMIT to define the number of possible
;               concurrent modified page writer I/O threads.
;               Add MPW_LOWAITLIMIT to define the modified page
;               writer busy wait low threshold.
;
;       X-17    HH0214          Hai Huang               27-Aug-1986
;               Set system date to 1986.
;
;       X-16    WMC0013         Wayne Cardoza           27-Aug-1986
;               Add second word tp PT free list.
;
;       X-15    WES0028         William E. Snaman       6-Aug-1986
;               Add CLU$GB_FALLBACK_MODE to enable fallback mode
;               during a rolling upgrade.
;
;       X-14    SJF             Stu Farnham             30-Jun-1986
;               Resolve conflicts from merge og SMP into main line
;
;       X-12    WMC0012         Wayne Cardoza           13-May-1986
;               Remove BOO$GL_SPTFREx.
;
;       X-10E3  HH0167          Hai Huang               07-Apr-1986
;               Relsove merge conflicts. Also, make the computed-value
;               area page-aligned for both SYSPARAM and PARAMETER.
;
;       X-10    ROW0533         Ralph O. Weber          28-DEC-1985
;               Comment out MAXATTCHPRI definition.  New system parameters
;               cannot be added until SYS.EXE is rebuilt.
;
```

```
514 ;       X-9     HWS0216         Harold Schultz          20-Dec-1985
515 ;               Change defaults for PIOPAGES (180 -> 245) and
516 ;               MAXQUEPRI (200 -> 100).
517 ;
518 ;       X-8                     Stu Farnham             19-Dec-1985
519 ;               Add MAXATTACHPRI, Maximum process priority to schedule
520 ;               on ASMP attached processor.
521 ;
522 ;       X-7                     Harold Schultz          15-Nov-1985
523 ;               Change CLISYMTBL maximum from 128 to 500. Also change
524 ;               default value from 100 to 250.
525 ;
526 ;       X-6                     Michael S. Harvey       14-Nov-1985
527 ;               Bring over change from V4.3 that is related to the
528 ;               merging of the various flavors of SYSBOOT.
529 ;
530 ;               LJK4014         Lawrence J. Kenah       21-Aug-1985
531 ;               Change maximum value of GBLSECTIONS from -1 (unbounded)
532 ;               to 4095.
533 ;
534 ;       X-5     PLL0002         Pamela Levesque         14-Nov-1985
535 ;               Remove qdss console bit from EXE$GL_WSFLAGS, as it
536 ;               is not being used.
537 ;
538 ;       X-6D1   SF04001         Stephen Fiorelli        11-Dec-1985
539 ;               Resolve conflicts from initial merge of exec reorg
540 ;               thread and mainline (4.4 BL7).
541 ;
542 ;       X-1C6   TCM0003         Trudy C. Matthews       03-Dec-1985
543 ;               For EXEC reorg project, increase the default number of
544 ;               interrupt stack pages to 4.
545 ;
546 ;       X-4     HWS0194         Harold Schultz          04-Oct-1985
547 ;               Change default value for MAXQUEPRI from 4 to 200
548 ;               Change default value for DEFQUEPRI from 4 to 100
549 ;               Change default value for PIOPAGES from 120 to 180
550 ;               Change default value for CLISYMTBL from 60 to 100
551 ;
552 ;       X-3     PLL0001         Pamela Levesque         17-Sep-1985
553 ;               Define a bit in EXE$GL_WSFLAGS to mean the console device
554 ;               is QDSS.
555 ;
556 ;       V04-001 RNH0011         Richard N. Holstein     04-Jun-1985
557 ;               Set the max for SRPCOUNT up to the max of SRPCOUNTV.
558 ;
559 ;       V03-080 DWT0238         David W. Thiel          24-Aug-1984
560 ;               Change default value of QDSKINTERVAL to 20 seconds.
561 ;
562 ;       V03-079 MIR0470         Michael I. Rosenblum    13-Aug-1984
563 ;               make tty_defport a specail sysgen parameter
564 ;
565 ;       V03-078 CDS0004         Christian D. Saether    06-Aug-1984
566 ;               Fix bug initializing exe$gl_static_flags
567 ;               introduced by cds0003.
568 ;
569 ;       V03-077 WMC0075         Wayne Cardoza           06-Aug-1984
570 ;               Make SRP default 96.
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 11
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  571 ;
  572 ;       V03-076 WMC0074         Wayne Cardoza           30-Jul-1984
  573 ;               Raise minimum working set parameters.
  574 ;
  575 ;       V03-075 DWT0231         David W. Thiel          25-Jul-1984
  576 ;               Raise RECNXINTERVAL to 60 seconds.
  577 ;
  578 ;       V03-074 DWT0230         David W. Thiel          25-Jul-1984
  579 ;               Raise RECNXINTERVAL to 20.  Make QDSKINTERVAL non-dynamic.
  580 ;
  581 ;       V03-073 WMC0073         Wayne Cardoza           23-Jul-1984
  582 ;               Raise max VIRTUALPAGECNT.
  583 ;
  584 ;       V03-072 BLS0334         Benn Schreiber          23-JUL-1984
  585 ;               Raise default ENQLM,ASTLM,DIOLM and BIOLM parameters.
  586 ;
  587 ;       V03-071 WHM0004         Bill Matthews           23-Jul-1984
  588 ;               Added a work station flag longword and defined the parameter
  589 ;               WS_OPA0. Made LGI_BRK_TERM default to true. Raised paging file
  590 ;               quota minimum from 256 to 512.
  591 ;
  592 ;       V03-070 CDS0003         Christian D. Saether    20-Jul-1984
  593 ;               Add one more buffer pool, ACP_DINDXCACHE, to file system caches.
  594 ;               Add ACP_REBLDSYSD flag.
  595 ;
  596 ;       V03-069 ACG0436         Andrew C. Goldstein,    12-Jul-1984  11:50
  597 ;               Add LGI_BRK_TERM and LGI_BRK_DISUSER parameters
  598 ;
  599 ;       V03-069 CDS0002         Christian D. Saether    11-July-1984
  600 ;               Raise minimum ACP_HDRCACHE to 3.
  601 ;               Change default for ACP_MULTIPLE to 0.
  602 ;
  603 ;       V03-068 JEJ0047         J E Johnson             06-Jul-1984
  604 ;               Change RMS_GBLBUFQUO to be a dynamic parameter.
  605 ;
  606 ;       V03-067 WMC0067         Wayne Cardoza           06-Jun-1984
  607 ;               Fix PQL minimum working set parameters.
  608 ;               Raise PHYSICALPAGES limits.
  609 ;
  610 ;       V03-066 HWS0069         Harold Schultz          24-May-1984
  611 ;               Change default size of CLISYMTBL from 40 to 60 pages.
  612 ;
  613 ;       V03-065 WHM0003         Bill Matthews           20-Apr-1984
  614 ;               Removed USESYSPARAMS. Use of the separate parameter file
  615 ;               is now required.
  616 ;
  617 ;       V03-064 MIR0400         Michael I. Rosenblum    10-Apr-1984
  618 ;               Add TTY_DEFPORT default port function longword.
  619 ;
  620 ;       V03-063 RAS0281         Ron Schaefer            09-Apr-1984
  621 ;               Add RMS_DFNBC default network block count parameter.
  622 ;
  623 ;       V03-062 WHM0002         Bill Matthews           04-Apr-1984
  624 ;               Added USESYSPARAMS and WRITESYSPARAMS to support the
  625 ;               default seperate system parameter file.
  626 ;               Changed the default for NPAGEDYN from 64000 to 131072(256 pages)
  627 ;               Added support for ascii sysgen parameters longer than 4 bytes.
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 12
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  628 ;               Replaced SCSNODEL and SCSNODEH with SCSNODE.
  629 ;               Replaced DISK_QUORUM1-4 with DISK_QUORUM.
  630 ;               Changed SAVEDUMP from type SPECIAL to type SYS.
  631 ;               Changed the units field for ACP_DATACHECK and ACP_SWPFLAGS
  632 ;               from boolean to Bit-mask.
  633 ;
  634 ;       V03-061 WMC0060         Wayne Cardoza           28-Mar-1984
  635 ;               Add MMG$GL_MAXMEM.
  636 ;
  637 ;       V03-060 JEJ0013         J E Johnson             25-Mar-1984
  638 ;               Add RMS_GBLBUFQUO sysgen parameter.
  639 ;
  640 ;       V03-059 WMC0059         Wayne Cardoza           24-Mar-1984
  641 ;               Add ACP_XQP_RES flag
  642 ;
  643 ;       V03-058 LMPBUILD        L. Mark Pilant,         19-Mar-1984  12:20
  644 ;               Make sure that the cells moved in LMP0205 are added with
  645 ;               the proper conditionals around them.
  646 ;
  647 ;       V03-057 LMP0205         L. Mark Pilant,         7-Mar-1984  11:23
  648 ;               Move EXE$GL_DYNAMIC_FLAGS and EXE$GL_STATIC_FLAGS from
  649 ;               SYSCOMMON.
  650 ;
  651 ;       V03-056 CDS0001         Christian D. Saether    28-Feb-1984
  652 ;               Raise default and min size of pagedyn to account for
  653 ;               xqp block caches.  Also SYSMWCNT.
  654 ;
  655 ;       V03-055 MMD0246         Meg Dumont,     27-Feb-1984  10:45
  656 ;               Add support for $MTACCESS installation specific accessibility
  657 ;               routine
  658 ;
  659 ;       V03-054 SSA0009         Stan Amway              13-Feb-1984
  660 ;               Changed default for PFRATL to 0.
  661 ;
  662 ;       V03-053 TMK0003         Todd M. Katz            02-Feb-1984
  663 ;               Change the name of the SCS parameter PAPORTPOLL to PANOPOLL.
  664 ;               Also change its address (from SCS$GB_PAPPOOL to SCS$GB_PANOPOLL)
  665 ;               and its default value (from 1 to 0).
  666 ;
  667 ;       V03-052 WHM0001         Bill Matthews           01-Feb-1984
  668 ;               Added new type for LGI_ parameters.
  669 ;
  670 ;       V03-051 TMK0002         Todd M. Katz            30-Jan-1984
  671 ;               Add the special parameters PE1,PE2,PE3,PE4,PE5,PE6 for use
  672 ;               by the PEDRIVER.
  673 ;
  674 ;               Add the SCS parameter PAPORTPOLL. If this boolean parameter is
  675 ;               set the local CI port(s) will poll remote ports. If it isn't it
  676 ;               won't.
  677 ;
  678 ;               Change the following SYSGEN parameters' DEFAULT, MAX, MIN etc..
  679 ;
  680 ;               1. Change the DEFAULT of MAXBUF from 1568 to 1584.
  681 ;               2. Change the DEFAULT of SCSSYSTEMID from 1 to 0.
  682 ;               3. Change the MIN of PAMAXPORT from -1 to 0.
  683 ;               4. Change the MAX of PAMAXPORT from -1 to 223.
  684 ;               5, Change the ADDRESS of PAMAXPORT from SCS$GW_PAMXPORT to
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 13
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

685 ;                   SCS$GB_PAMXPORT.
686 ;                6. Change the size of PAMAXPORT from WORD to BYTE.
687 ;
688 ;       V03-050 WMC0048         Wayne Cardoza           16-Jan-1984
689 ;               CJF related parameters must be made SPECIAL and defaulted
690 ;               off.
691 ;
692 ;       V03-049 JLV0328         Jake VanNoy             11-JAN-1984
693 ;               Add TTY_TIMEOUT and TTY_AUTOCHAR.
694 ;
695 ;       V03-048 KTA3097         Kerbey T. Altmann       10-Jan-1984
696 ;               Set date to 1984.
697 ;
698 ;       V03-047 WMC0047         Wayne Cardoza           19-Dec-1983
699 ;               WSMAX max is 64000 not 65280.
700 ;               Make MMG$GW_BIGPFN a new cell.
701 ;
702 ;       V03-046 LMP0177         L. Mark Pilant,         7-Dec-1983  11:32
703 ;               Add a dymanic parameter to control whether or not
704 ;               non-discretionary classification checks are to be performed.
705 ;
706 ;       V03-045 SSA0003         Stan Amway               5-Dec-1983
707 ;               Added DORMANTWAIT to support outswap scheduling changes.
708 ;               Changed units and default for LONGWAIT.
709 ;
710 ;       V03-044 DWT0150         David W. Thiel          18-Nov-1983
711 ;               Add LOCKDIRWT and QDSKVOTES parameters.  Deleted old
712 ;               VAXCLUSTER bit and define new VAXLCUSTER as a byte.
713 ;               Reorder cluster parameters.  Make ALLOCLASS non-dynamic.
714 ;               Change PAPOLLINTERVAL default from 15 to 5 seconds.
715 ;               Change SCSMAXMSG default from 96 to 112.
716 ;               Change SRPSIZE default from 96 to 128 to close hole
717 ;               between SRP and IRP allocation pending a more complete
718 ;               examination of pool allocation.
719 ;
720 ;       V03-043 TMK0001         Todd M. Katz            12-Oct-1983
721 ;               Add PQL_DJTQUOTA and PQL_MJTQUOTA - the default and minimum
722 ;               byte creation quotas for job-wide logical name tables.
723 ;
724 ;       V03-042 ACG0360         Andrew C. Goldstein,    21-Sep-1983  16:25
725 ;               Change defaults for LGI_xxx breakin parameters
726 ;
727 ;       V03-041 ACG0350         Andrew C. Goldstein,    19-Aug-1983  17:56
728 ;               Raise MAXBUF minimum to 1200 to allow BACKUP to work
729 ;
730 ;       V03-040 GAS0162         Gerry Smith             30-Jul-1983
731 ;               Add LGI_PWD_TMO, the system password drop dead time.
732 ;
733 ;       V03-039 NPK3030         N. Kronenberg           29-Jul-1983
734 ;               Change PAMAXPORT maximum from 223 to -1.
735 ;
736 ;       V03-038 RAS0179         Ron Schaefer            29-Jul-1983
737 ;               Delete LOGPHASHTBL,LOGGHASHTBL,LOGSHASHTBL.
738 ;               Change default of IMGIOCNT from 32 to 64.
739 ;               Change default of RMS_EXTEND from 80 to 0.
740 ;
741 ;       V03-037 KFH0004         Ken Henderson           28 Jul 1983
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 14
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  742 ;               Add MAXQUEPRI, DEFQUEPRI, QDSKINTERVAL
  743 ;               Remove JOBQUEUES, REINITQUE, MAXPRINTSYMB
  744 ;               Change VMS_CLUSTER to VAXCLUSTER
  745 ;               Modify default of RECNXINTERVAL from 3 to 10
  746 ;
  747 ;       V03-036 NPK3029         N. Kronenberg           26-Jul-1983
  748 ;               Tune up the SCS and PA parameters.  Remove PASTRETRY.
  749 ;               Add PAMXPORT, PASANITY.
  750 ;
  751 ;       V03-035 MSH0002         Maryann Hinden          08-Jul-1983
  752 ;               Add cluster quorum disk parameters.
  753 ;
  754 ;       V03-034 GAS0142         Gerry Smith             20-Jun-1983
  755 ;               Add the login security parameters.
  756 ;
  757 ;       V03-033 PCA1015         Paul C. Anagnostopoulos 13-Jun-1983
  758 ;               Fix the valid ranges for the STARTUP_Pn parameters.
  759 ;
  760 ;       V03-032 TCM0002         Trudy C. Matthews       1-Jun-1983
  761 ;               Add ALLOCLASS parameter, a cluster parameter that defines
  762 ;               the device allocation class for this system.
  763 ;
  764 ;       V03-031 SRB0086         Steve Beckhardt         24-May-1983
  765 ;               Made LOCKIDTBL_MAX dynamic.
  766 ;
  767 ;       V03-030 JSV0295         Joost Verhofstad        20-MAY-1983
  768 ;               Add SYSGEN parameters CJFLOAD and CJFSYSRUJ
  769 ;
  770 ;       V03-029 KFH0003         Ken Henderson           20 May 1983
  771 ;               Increased default values:
  772 ;               SPTREQ          720 -> 896
  773 ;               GBLPAGES        3072 -> 4096
  774 ;               GBLSECTIONS     80 -> 128
  775 ;
  776 ;       V03-028 KDM0044         Kathleen D. Morse       03-May-1983
  777 ;               Add EXE$GL_ARCHFLAG.
  778 ;
  779 ;       V03-027 PCA1015         Paul C. Anagnostopoulos 28-Apr-1983
  780 ;               Add TAILORED parameter to specify whether or not this system
  781 ;               is tailored (has a library disk).
  782 ;               Add STARTUP_Pn parameters for passing information to the
  783 ;               system startup procedure.
  784 ;
  785 ;       V03-026 SRB0081         Steve Beckhardt         28-Apr-1983
  786 ;               Added new parameter LOCKIDTBL_MAX.
  787 ;
  788 ;       V03-025 RNG0025         Rod Gamache             21-Apr-1983
  789 ;               Change default value of MAXBUF to be more flexible
  790 ;               for the DEUNA device driver.
  791 ;
  792 ;       V03-024 KFH0002         Ken Henderson           14 Apr 1983
  793 ;               Modify call to SYI_ITEM_CODE to
  794 ;               support Ascii sysnodename.
  795 ;
  796 ;       V03-023 MIR0030         Michael I. Rosenblunm   14-Apr-1983
  797 ;               Make line editing the default
  798 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 15
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  799 ;       V03-022 TCM0001         Trudy C. Matthews       8-Apr-1983
  800 ;               Allow WSMAX to grow to 65280.
  801 ;
  802 ;       V03-021 MSH0001         Maryann Hinden          25-Mar-1983
  803 ;               Add ASCII type.  Correct default for SCSNODEH.
  804 ;
  805 ;       V03-020 DWT0080         David W. Thiel          1-Mar-1983
  806 ;               Define cluster class of parameters containing
  807 ;               QUORUM, VOTES, and CNXRETRYINT.
  808 ;
  809 ;       V03-019 JWH0191         Jeffrey W. Horn         28-Feb-1983
  810 ;               Change default value for PIOPAGES.
  811 ;
  812 ;       V03-018 KFH0001         Ken Henderson           15 Feb 1983
  813 ;               Added conditionals for GETSYISW, to let
  814 ;               this file be used to define the sysboot
  815 ;               parameters for SYS$GETSYI/F$GETSYI/LIB$GETSYI
  816 ;               ******************************************************
  817 ;               ONE SIDE EFFECT OF THIS MOD IS TO REQUIRE THAT ALL
  818 ;               ALLOCATION OF MEMORY CELLS BE CONDITIONALIZED TO NOT
  819 ;               HAPPEN IF GETSYISW IS DEFINED.
  820 ;               ******************************************************
  821 ;
  822 ;       V03-017 RNG0017         Rod N. Gamache          11-Feb-1983
  823 ;               Change default value for PHYSICALPAGES.
  824 ;
  825 ;       V03-016 DWT0071         David W. Thiel          28-Jan-1983
  826 ;               Add VMS_CLUSTER parameter and corresponding
  827 ;               LOADCLUSTR bit in SGN$GL_LOADFLAGS.
  828 ;               Add PRCPOLINTERVAL parameter with global name
  829 ;               SCS$GW_PRCPOLINT.
  830 ;
  831 ;       V03-015 STJ3053         Steven T. Jeffreys      21-Jan-1983
  832 ;               Added LOADERAPAT and LOADCHKPRT parameters, and defined
  833 ;               SGN$GL_LOADFLAGS, a system global longword to control
  834 ;               the loading of various pieces of the EXEC.
  835 ;
  836 ;       V03-014 KTA3029         Kerbey T. Altmann       11-Jan-1983
  837 ;               Set date to 1983.
  838 ;
  839 ;       V03-013 SRB0057         Steve Beckhardt         16-Dec-1982
  840 ;               Increased maximum size of LOCKIDTBL to 65535.
  841 ;
  842 ;       V03-012 DMW4016         DMWalp                  15-Dec-1982
  843 ;               Added parameters for size of new logical name hash tables
  844 ;
  845 ;       V03-011 JWH0117         Jeffrey W. Horn         29-Oct-1982
  846 ;               Add PIOPAGES CTLPAGES, and CTLIMGLIM.
  847 ;               Change maximum of PHYSICALPAGES to 65536.
  848 ;               Change default of SCSSYSTEMIDH to 0.
  849 ;
  850 ;       V03-010 KTA3016         Kerbey T. Altmann       21-Oct-1982
  851 ;               Add SCSNODENAME.
  852 ;
  853 ;       V03-009 HRJ0064         Herb Jacobs             21-Apr-1982
  854 ;               Fix default values of RMS_DFMBC, RMS_EXTEND, MPW_LOLIMIT.
  855 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 16
X-101U18 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  856 ;       V03-008 JLV0208         Jake Vannoy             15-Apr-1982
  857 ;               Fix default values for TTY_SILOTIME, WSINC, WSDEC.
  858 ;
  859 ;       V03-007 JLV0207         Jake VanNoy              5-APR-1982
  860 ;               Add some smarts to PARAMETER macro to ignore dynamic
  861 ;               bits if they are not in EXE$GL_DEFFLAGS. This prevents
  862 ;               the dynamic bits in STJ0249 from being included in
  863 ;               PRM$M_DYNFLAGS.
  864 ;
  865 ;       V03-006 STJ0249         Steven T. Jeffreys      01-Apr-1982
  866 ;               Add global longword for system message flags.  Define
  867 ;               EXE$V_MOUNTMSG and EXE$V_DISMOUMSG to control operator
  868 ;               notification of mounts and dismounts, respectively.
  869 ;               By default, both are disabled.
  870 ;
  871 ;       V03-005 MLJ0085         Martin L. Jack          01-Apr-1982
  872 ;               Add EXE$V_JOBQUEUES, EXE$V_REINITQUE to control initialization
  873 ;               of JBCSYSQUE by job controller.
  874 ;
  875 ;       V03-004 PHL0041         Peter H. Lipman         01-Apr-1982
  876 ;               Default setting for SAVEDUMP must be off.
  877 ;
  878 ;       V03-003 HRJ0061         Herb Jacobs             28-Mar-1982
  879 ;               Fix categories for /MAJOR, /SYS, /SYSGEN, correct
  880 ;               some default values, and change global name of SWPFILCNT.
  881 ;
  882 ;       V03-002 PHL0040         Peter H. Lipman         22-Mar-1982
  883 ;               Add EXE$V_PAGFILDMP, EXE$V_SAVEDUMP, EXE$GW_PGFL_FID
  884 ;               to support the dump file in the page file.
  885 ;
  886 ;       V03-001 JLV0193         Jake VanNoy             15-MAR-1982
  887 ;               Add TTY_SILOTIME. Change defaults for TTY_OWNER and TTY_PROT
  888 ;               and PQL_ENQLM. Change names of TTYSCANDELTA to TTY_SCANDELTA
  889 ;               and DIALTYPE to TTY_DIALTYPE.
  890 ;
  891 ;--
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 17
X-101U18 DECLARATIONS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  893          .SBTTL  DECLARATIONS
  894 ;
  895 ; INCLUDE FILES:
  896 ;
  897          $BPTDEF                         ; DEFINE INITIAL BREAKPOINT CALLERS
  898          $SYIDEF                         ; DEFINE SYS$GETSYI ITEM CODES
  899          $PQLDEF                         ; DEFINE QUOTA LIST CODES
  900          $PRMDEF                         ; DEFINE PARAMETER DESCRIPTOR
  901          $SGNDEF                         ; SYSGEN CONSTANTS
  902          $SYSVERSIONDEF                  ; Version bit definitions

      ;
      ;
      ;               DEFINITIONS OF VERSION CATEGORIES
      ;
      ;
      ; BASE_IMAGE
      ;
      ;       Base image transfer vectors.  To be changed whenever a transfer vector
      ;       is added.  The major ID is to be changed only when there is no
      ;       alternative.  This will force all code to relink.
      ;
      ;
      ; MEMORY_MANAGEMENT
      ;
      ;       All memory management data structures and routines.  This is also used
      ;       for pool management.
      ;
      ;
      ; IO
      ;
      ;       All I/O  data structures and routines.
      ;
      ;
      ; FILES_VOLUMES
      ;
      ;       All RMS and file system related items.
      ;
      ;
      ; PROCESS_SCHED
      ;
      ;       Data structures  and routines related to process control, scheduling,
      ;       and process structure.   This includes the layout of P1 space.
      ;       Timers, ASTs and event flags are also included in this category.
      ;
      ;
      ; SYSGEN
      ;
      ;       SYSGEN parameters.  This category should not be used for any other
      ;       purpose.  Its intent is to allow us to eventually move all SYSGEN
      ;       parameters to another virtual address range.
      ;
      ;
      ; CLUSTERS_LOCKMGR
      ;       Routines and data structures related to clusters, connection manager,
      ;       lock manager, and similar cluster wide facilities.
      ;
```

```
;
; LOGICAL_NAMES
;
;       Logical name  data structures and routines.
;
;
; SECURITY
;
;       The security subsystem.
;
;
; IMAGE_ACTIVATOR
;
;       Image activation and image file interpretation.
;
;
; NETWORKS
;
;       DECnet and support for datalink drivers.
;
;
; COUNTERS
;
;       Cells which are interpreted as counts.  This category is separate
;       because it should almost never be necessary to change the major ID
;       of this category.
;
;
; STABLE
;
;       Routines and data  structures which are expected to be very stable.
;       If a change is necessary, the cell or routine should be renamed rather
;       than change the major ID.  The major ID may be changed  every few years
;       to garbage collect obsolete cells.
;
;
; MISC
;
;       Things that don't fit any other category.
;
;
; CPU
;
;       CPU  specific support.
;
;
; VOLATILE
;
;       The major ID of this category will be iuncremented on a fairly regular
;       basis.  This should be used for new items which are expected to undergo
;       a large amount of change.  If for example, a new IO structure is added,
;       but is expected to have significant change in the next release, it
;       should be categorized as both volatile and IO.  In this way, it will
;       not be necessary to change the major ID for the IO category when
;       the structure changes.  Thus, not all drivers will be forced to relink.
;
;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 19
X-101U18 DECLARATIONS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

                ; SHELL
                ;
                ;       Cells related to the layout of SHELL.
                ;
                ;
903                     $TTDEF                          ; DEFINE TERMINAL CHARACTERISTICS
904                     $TT2DEF                         ; DEFINE MORE TERMINAL DEFINTIONS
905
906             ;
907             ; MACRO TO GENERATE PARAMETER DESCRIPTOR IF PRMSW IS TRUE OTHERWISE
908             ; SIMPLY DEFINE PARAMETERS
909             ;
910             ;
911             ; MACROS:
912             ;
913
914                     .MACRO  PFNALC  SIZ,SYMLST,VERSION_MASK
915                     .IF     NOT_DEFINED     VERSION
916                     .ALIGN  LONG
917                     .ENDC
918                     .IRP    SYM,<SYMLST>
919                     DEFINE  PFN$A'SIZ'_'SYM,VERSION_MASK
920                     .ENDR
921                     .LONG   0
922                     .ENDM   PFNALC
923
924                     .MACRO  PARAMETER,ADDRESS,NAME,TYPE=STATIC,DEFAULT=0,MIN=-1,MAX=-1,-
925                             UNIT,SIZE=LONG,BIT,VERSION_MASK
926
927             ;
928             ;       If we are define version masks, the only thing we assemble are the version
929             ;       masks associated with the system parameter
930             ;
931
932                     .IF     DEFINED,VERSION
933                     ADDRESS=0
934                     .IRP    ...subversion...,<VERSION_MASK>
935                             ADDRESS==ADDRESS!<1@SYS$K_'...subversion...>
936                     .ENDR
937                     .IF_FALSE
938
939             ;
940             ; When GETSYISW is defined, the macro PARAMETER becomes a conduit to
941             ; a lower level macro SYI_ITEM_CODE. In the fashion of JPI_ITEM_CODE and
942             ; DVI_ITEM_CODE, SYI_ITEM_CODE is called multiple times (once per item)
943             ; by the larger macro SYI_GENERATE_TABLE. This file becomes the definition
944             ; of SYI_GENERATE_TABLE when GETSYISW is defined.
945             ;
946                     OUTLEN = 4
947                     .IIF    IDENTICAL <SIZE><BYTE>, OUTLEN = 1
948                     .IIF    IDENTICAL <SIZE><WORD>, OUTLEN = 2
949                     .IIF    IDENTICAL <SIZE><LONG>, OUTLEN = 4
950                     .IIF    IDENTICAL <SIZE><QUAD>, OUTLEN = 8
951                     .IIF    IDENTICAL <SIZE><OCTA>, OUTLEN = 16
952
953                     .IF     DEFINED GETSYISW
954
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 20
X-101U18 DECLARATIONS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

  955          .IF     BLANK BIT
  956
  957          .IF     NOT_DEFINED SYI$_'NAME
  958          .WARN ; SYI$_'NAME SHOULD BE DEFINED IN STARDEFQZ.SDL (EXE-SECT)
  959          .ENDC
  960
  961          .IF     IDENTICAL <UNIT><Ascii>
  962
  963          SYI_ITEM_CODE   EXE,-           ; TABLE BASE - EXEC CELLS
  964                          <NAME>,-        ; ITEM NAME
  965                          <ADDRESS>,-     ; SOURCE
  966                          PADSTR,-        ; DTYPE = PADDED STRING
  967                          0,-             ; BITPOS
  968                          PRM$C_'SIZE',-  ; BITSIZ
  969                          OUTLEN          ; OUTLEN
  970
  971          .IF_FALSE
  972
  973          SYI_ITEM_CODE   EXE,-           ; TABLE BASE - EXEC CELLS
  974                          <NAME>,-        ; ITEM NAME
  975                          <ADDRESS>,-     ; SOURCE
  976                          DECNUM,-        ; DTYPE = DECIMAL NUMBER
  977                          0,-             ; BITPOS
  978                          PRM$C_'SIZE',-  ; BITSIZ
  979                          OUTLEN          ; OUTLEN
  980
  981          .ENDC           ; IDENTICAL <UNIT><Ascii>
  982
  983          .IF_FALSE       ; BLANK BIT
  984
  985          .IF     NOT_DEFINED SYI$_'NAME
  986          .WARN ; SYI$_'NAME SHOULD BE DEFINED IN STARDEFQZ.SDL (FLD-SECT)
  987          .ENDC
  988
  989          SYI_ITEM_CODE   FLD,-           ; TABLE BASE - FIELD DATA
  990                          <NAME>,-
  991                          <ADDRESS>,-
  992                          BITVAL,-
  993                          <BIT>,-
  994                          1,-
  995                          1
  996
  997          .ENDC           ; BLANK BIT
  998
  999          .IF_FALSE       ; DEFINED GETSYISW
 1000
 1001          .IF     DF,PRMSW                ; DO IF CREATING PARAMETER DESCRIPTOR
 1002 PRMSAV...=.                              ; SAVE LOC COUNTER
 1003          .PSECT  $$$918,LONG             ;
 1004          BAS...=.                        ; SET BASE FOR THIS DESCRIPTOR
 1005          .BLKB   PRM$C_LENGTH            ; GENERATE SPACE
 1006          SAV...=.                        ;
 1007          PRM     L_ADDR                  ;
 1008          .LONG   ADDRESS                 ;
 1009          PRM     L_DEFAULT               ;
 1010          .LONG   DEFAULT                 ;
 1011          PRM     L_MIN                   ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 21
X-101U18 DECLARATIONS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 1012          .LONG   MIN                     ;
 1013          PRM     L_MAX                   ;
 1014          .LONG   MAX                     ;
 1015          .IF     GREATER         <%LENGTH(NAME)-PRM$C_MAXNAMLEN>
 1016          .ERROR  ; The parameter called NAME has too many characters
 1017          .ENDC
 1018          PRM     T_NAME                  ;
 1019          .ASCIC  %'NAME'%                ;
 1020          .IF     GREATER         <%LENGTH(UNIT)-PRM$C_MAXUNILEN>
 1021          .ERROR  ; The quantity called UNIT has too many characters
 1022          .ENDC
 1023          PRM     T_UNIT                  ;
 1024          .ASCIC  %'UNIT'%                ;
 1025          PRM     B_SIZE                  ; SET FIELD SIZE
 1026          .IF     B,BIT                   ;
 1027          .BYTE   PRM$C_'SIZE             ;
 1028          .IFF                            ;
 1029          .BYTE   1                       ;
 1030          PRM     B_POS                   ;
 1031          .BYTE   BIT                     ;
 1032          .ENDC                           ;
 1033          PRM     L_FLAGS                 ;
 1034          TYP...=0                        ;
 1035          .IRP    TYPNAM,<TYPE>           ;
 1036          TYP...=TYP...!PRM$M_'TYPNAM     ;
 1037
 1038          .IF NB BIT                      ; DEFINE PRM$M_DYNFLAGS
 1039          .IF EQ PRM$M_'TYPNAM-PRM$M_DYNAMIC
 1040          .IF IDN ADDRESS,EXE$GL_DEFFLAGS
 1041          PRM$M_DYNFLAGS == PRM$M_DYNFLAGS!<1@BIT>
 1042          .ENDC
 1043          .ENDC
 1044          .ENDC
 1045
 1046          .ENDR                           ;
 1047          .LONG   TYP...                  ;
 1048          .=SAV...                        ; REPOSITION LOCATION COUNTER
 1049          .PSECT  $$$917A,PAGE            ; BACK TO NORMAL PSECT
 1050          .=PRMSAV...                     ; RESTORE LOCATION COUNTER
 1051          .IFF                            ;
 1052          .IF     B,BIT                   ;
 1053          .IF     GT,OUTLEN-4             ; IF LENGTH >4 FORCE LONGWORD ALIGNMENT
 1054          .ALIGN  LONG                    ;
 1055          .IFF
 1056          .ALIGN  'SIZE'                  ;
 1057          .ENDC
 1058 ADDRESS'::                               ; DEFINE GLOBAL VALUE
 1059          .ENDC                           ;
 1060          .ENDC                           ;
 1061          .IF     B,BIT                   ;
 1062          .IF     GT,OUTLEN-4             ; IF LENGTH >4 FORCE LONGWORD ALIGNMENT
 1063          .ALIGN  LONG                    ;
 1064          .LONG   DEFAULT                 ; GENERATE DEFAULT QUAD VALUE
 1065          .LONG   DEFAULT                 ;
 1066          .IF     GT,OUTLEN-8             ;
 1067          .LONG   DEFAULT                 ; GENERATE DEFAULT OCTA VALUE
 1068          .LONG   DEFAULT                 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 22
X-101U18 DECLARATIONS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

1069          .ENDC                           ;
1070          .IFF                            ;
1071          .ALIGN  'SIZE'                  ;
1072          .'SIZE' DEFAULT                 ; GENERATE DEFAULT VALUE
1073          .ENDC
1074          .ENDC                           ;
1075
1076          .ENDC           ; DEFINED GETSYISW
1077
1078          .ENDC           ; Defined version
1079          .ENDM   PARAMETER               ;
1080
1081          .MACRO  PRM,OFFSET,VERSION_MASK
1082          .IF     DEFINED,VERSION
1083          DEFINE  PRM$'OFFSET,VERSION_MASK
1084          .IF_FALSE
1085          .IF     NOT_DEFINED GETSYISW
1086          .=BAS...+PRM$'OFFSET            ;
1087          .ENDC
1088          .ENDC
1089          .ENDM   PRM                     ;
1090
1091 ;
1092 ; MACRO TO CONDITIONALLY DEFINE LABELS
1093 ;
1094          .MACRO  DEFINE,LABEL,VERSION_MASK
1095          .IF     DEFINED,VERSION
1096          LABEL=0
1097          .IRP    ...subversion...,<VERSION_MASK>
1098                  LABEL==LABEL!<1@SYS$K_'...subversion...>
1099          .ENDR
1100          .IF_FALSE
1101          .IF     NOT_DEFINED GETSYISW
1102          .IF     NDF,PRMSW               ;
1103 LABEL'::                                 ;
1104          .IFF                            ;
1105 LABEL':                                  ;
1106          .ENDC                           ;
1107          .ENDC   ; NOT_DEFINED GETSYISW
1108          .ENDC   ; Version
1109          .ENDM   DEFINE                  ;
1110 ;
1111 ;        MACRO TO GENERATE PROCESS QUOTA LIST TABLES
1112 ;
1113 ;        PQL     QUOTA_NAME,DEFAULT,MIN,FLAG,UNIT,DYNAMIC_STATE
1114 ;
1115          .MACRO  PQL Q,DEFLT=0,MINIM=0,FLAG=0,UNT,DYNAMIC_FLAG=DYNAMIC,PQL_VERSION_MA
1116          .IF     NOT_DEFINED VERSION
1117          .IF     NOT_DEFINED GETSYISW
1118          .IF     NDF,PRMSW               ;FOR LINKAGE INTO BASE SYS...
1119          .PSECT  $$$$$SYSPARAM_DATA,PAGE,WRT,NOEXE,OVR
1120          .IFF                            ;
1121          .PSECT  $$$917A,PAGE            ;
1122          .ENDC                           ;
1123          PQLSAV...=.                     ; SAVE LOCATION COUNTER
1124          .=PQL$AL_DEFAULT+<4*PQL$_'Q>    ; POINT INTO DEFAULT TABLE
1125          .ENDC   ; NOT_DEFINED GETSYISW
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 23
X-101U18 DECLARATIONS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 1126          .ENDC
 1127          PARAMETER       ADDRESS=PQL$GD'Q,-      ;
 1128                          DEFAULT=DEFLT,-         ;
 1129                          NAME=PQL_D'Q,-  ;
 1130                          SIZE=LONG,-             ;
 1131                          TYPE=<PQL,SYSGEN,DYNAMIC_FLAG>,-        ;
 1132                          UNIT=UNT,-
 1133                          VERSION_MASK=PQL_VERSION_MASK
 1134          .IF     NOT_DEFINED VERSION
 1135          .IF     NOT_DEFINED GETSYISW
 1136          .IF     NDF,PRMSW               ;FOR LINKAGE WITH BASE SYS...
 1137          .PSECT  $$$$$SYSPARAM_DATA,PAGE,WRT,NOEXE,OVR
 1138          .IFF
 1139          .PSECT  $$$917A,PAGE            ;
 1140          .ENDC                           ;
 1141 PQLSAV...=.                              ;
 1142          .=PQL$AL_MIN+<4*PQL$_'Q>        ; POINT INTO MINIMUM VALUE TABLE
 1143          .ENDC   ; NOT_DEFINED GETSYISW
 1144          .ENDC
 1145          PARAMETER       ADDRESS=PQL$GM'Q,-      ;
 1146                          DEFAULT=MINIM,-         ;
 1147                          NAME=PQL_M'Q,-  ;
 1148                          SIZE=LONG,-             ;
 1149                          TYPE=<PQL,SYSGEN,DYNAMIC_FLAG>,-        ;
 1150                          UNIT=UNT,-
 1151                          VERSION_MASK=PQL_VERSION_MASK
 1152          .IF     NOT_DEFINED VERSION
 1153          .IF     NOT_DEFINED GETSYISW
 1154          .IF     NDF,PRMSW
 1155          .PSECT  $$$$$SYSPARAM_DATA,PAGE,WRT,NOEXE,OVR
 1156          .IFF
 1157          .PSECT  $$$917A,PAGE
 1158          .ENDC                           ;
 1159 PQLSAV...=.                              ;
 1160          .=PQL$AB_FLAG+PQL$_'Q           ; POINT INTO FLAG BYTE FOR QUOTA
 1161          .BYTE   FLAG                    ; AND FILL IN FLAG
 1162          .=PQLSAV...                     ; RESTORE LOCATION COUNTER
 1163          .ENDC   ; NOT_DEFINED GETSYISW
 1164          .ENDC   ; Version
 1165          .ENDM   PQL                     ;
 1166
 1167 ;
 1168 ; EQUATED SYMBOLS:
 1169 ;
 1170 PQL_V_DEDUCT=0                           ; DEDUCTIBILE QUOTA FLAG
 1171
 1172 PQL_M_DEDUCT=1                           ; FLAG VALUE FOR DEDUCTIBLE QUOTA
 1173 DEDUCTIBLE=PQL_M_DEDUCT                  ; NAME FOR READIBILITY
 1174
 1175 .IF     DF,PRMSW                        ; DO IF PARAMETER
 1176 PRM$M_DYNFLAGS == 0
 1177 .ENDC
 1178
 1179          .IF     NOT_DEFINED VERSION
 1180 ;
 1181 ; DEFINE THE SYSTEM CONTROL FLAGS.  ANY FLAGS ADDED SHOULD BE PLACED
 1182 ; IN THE FIELD DEFINITION THAT CORRESPONDS TO THE CELL IN SYSCOMMON FROM
```

```
1183 ; WHICH THE FLAGS ARE REFERENCED.  (THIS WAS ORIGINALLY EXE$GL_FLAGS, BUT
1184 ; OVER TIME WILL SPLIT INTO EXE$GL_DYNAMIC_FLAGS, EXE$GL_STATIC_FLAGS, AND
1185 ; EXE$GL_STATE_FLAGS.)
1186 ;
1187         $GBLINI GLOBAL
1188         $VIELD  EXE,0,<-                ; DEFINITION FOR EXE$GL_FLAGS
1189                 SYSWRTABL,-             ; LEAVE SYSTEM READ ONLY CODE WRITABLE
1190                 NOAUTOCNF,-             ; NO AUTOMATIC CONFIGURATION OF UBA
1191                 POOLPGING,-             ; ENABLE DYNAMIC POOL PAGING
1192                 SIMULATOR,-             ; RUNNING ON SIMULATOR
1193                 CRDENABL,-              ; ENABLE CRD ERROR DETECTION
1194                 SBIERR,-                ; ENABLE SBI ERROR INTERRUPT
1195                 INIT,-                  ; RMS AND FILE SYSTEM INITIALIZED
1196                 SETTIME,-               ; FORCE SOLICITATION OF TIME
1197                 FATAL_BUG,-             ; FORCE ALL BUG CHECKS FATAL
1198                 MULTACP,-               ; USE MULTIPLE FILE ACP'S
1199                 NOCLUSTER,-             ; TURN OFF PAGE FAULT CLUSTERING
1200                 BUGREBOOT,-             ; AUTO REBOOT ON BUGCHECK
1201                 SYSUAFALT,-             ; ALTERNATE LOGICAL NAME FOR SYSUAF
1202                 SHRF11ACP,-             ; MAKE F11ACP SHARABLE AT BOOT TIME
1203                 BUGDUMP,-               ; TAKE SYSTEM DUMP ON BUGCHECK
1204                 RESALLOC,-              ; ENABLE RESOURCE ALLOCATION CHECKS
1205                 CONCEALED,-             ; ENABLE USE OF CONCEALED DEVICES
1206                 SSINHIBIT,-             ; INHIBIT SYSTEM SERVICES PER-PROCESS
1207                 EXPLICITP,-             ; IF SET TODAY IS CONSIDERED PRIMARY
1208                 EXPLICITS,-             ; IF SET TODAY IS CONSIDERED SECONDARY
1209                 PGFLFRAG,-              ; SET IF PAGE FILE FRAGMENTED MSG ISSUED
1210                 PGFLCRIT,-              ; SET IF PAGE FILE FULL MSG ISSUED
1211                 TBCHK,-                 ; SET IF PROCESSOR REGISTER TBCHK PRESENT
1212                 PAGFILDMP,-             ; SET IF DUMP IS IN PAGE FILE
1213                 SAVEDUMP,-              ; SET TO SAVE DUMP UNTIL ANALYZED
1214                 JOBQUEUES,-             ; Set if JOBCTL to enable queues
1215                 REINITQUE,-             ; Set if JOBCTL to reinitialize JBCSYSQUE
1216         >
1217 ;       $VIELD  EXE,0,<-                ; DEFINITION FOR EXE$GL_STATE_FLAGS
1218 ;       >
1219         $VIELD  EXE,0,<-                ; DEFINITION FOR EXE$GL_TIME_CONTROL
1220                 <NOCLOCK,1,M>,-         ; DO NOT TURN ON CLOCK
1221                 <NOSMPSANITY,1,M>,-     ; DISABLE SMP SANITY TIMER TIMEOUTS
1222                 <NOSPINWAIT,1,M>,-      ; DISABLE SMP SPIN/BUSYWAIT TIMEOUTS
1223         >
1224         .IF_FALSE
1225         .PSECT  $ABS$,ABS
1226
1227         DEFINE  EXE$V_SYSWRTABL,-
1228                 VERSION_MASK=<SYSGEN>
1229
1230         DEFINE  EXE$V_NOAUTOCNF,-
1231                 VERSION_MASK=<SYSGEN>
1232
1233         DEFINE  EXE$V_POOLPGING,-
1234                 VERSION_MASK=<SYSGEN>
1235
1236         DEFINE  EXE$V_SIMULATOR,-
1237                 VERSION_MASK=<SYSGEN>
1238
1239         DEFINE  EXE$V_CRDENABL,-
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 25
X-101U18 DECLARATIONS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

1240                    VERSION_MASK=<SYSGEN>
1241
1242            DEFINE  EXE$V_SBIERR,-
1243                    VERSION_MASK=<SYSGEN>
1244
1245            DEFINE  EXE$V_INIT,-
1246                    VERSION_MASK=<SYSGEN>
1247
1248            DEFINE  EXE$V_SETTIME,-
1249                    VERSION_MASK=<SYSGEN>
1250
1251            DEFINE  EXE$V_FATAL_BUG,-
1252                    VERSION_MASK=<SYSGEN>
1253
1254            DEFINE  EXE$V_MULTACP,-
1255                    VERSION_MASK=<SYSGEN>
1256
1257            DEFINE  EXE$V_NOCLUSTER,-
1258                    VERSION_MASK=<SYSGEN>
1259
1260            DEFINE  EXE$V_BUGREBOOT,-
1261                    VERSION_MASK=<SYSGEN>
1262
1263            DEFINE  EXE$V_SYSUAFALT,-
1264                    VERSION_MASK=<SYSGEN>
1265
1266            DEFINE  EXE$V_SHRF11ACP,-
1267                    VERSION_MASK=<SYSGEN>
1268
1269            DEFINE  EXE$V_BUGDUMP,-
1270                    VERSION_MASK=<SYSGEN>
1271
1272            DEFINE  EXE$V_RESALLOC,-
1273                    VERSION_MASK=<SYSGEN>
1274
1275            DEFINE  EXE$V_CONCEALED,-
1276                    VERSION_MASK=<SYSGEN>
1277
1278            DEFINE  EXE$V_SSINHIBIT,-
1279                    VERSION_MASK=<SYSGEN>
1280
1281            DEFINE  EXE$V_EXPLICITP,-
1282                    VERSION_MASK=<SYSGEN>
1283
1284            DEFINE  EXE$V_EXPLICITS,-
1285                    VERSION_MASK=<SYSGEN>
1286
1287            DEFINE  EXE$V_PGFLFRAG,-
1288                    VERSION_MASK=<SYSGEN>
1289
1290            DEFINE  EXE$V_PGFLCRIT,-
1291                    VERSION_MASK=<SYSGEN>
1292
1293            DEFINE  EXE$V_TBCHK,-
1294                    VERSION_MASK=<SYSGEN>
1295
1296            DEFINE  EXE$V_PAGFILDMP,-
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 26
X-101U18 DECLARATIONS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

                        VERSION_MASK=<SYSGEN>

                DEFINE  EXE$V_SAVEDUMP,-
                        VERSION_MASK=<SYSGEN>

                DEFINE  EXE$V_JOBQUEUES,-
                        VERSION_MASK=<SYSGEN>

                DEFINE  EXE$V_REINITQUE,-
                        VERSION_MASK=<SYSGEN>

                DEFINE  EXE$V_SA_APP,-
                        VERSION_MASK=<SYSGEN>

                DEFINE  EXE$M_NOCLOCK,-
                        VERSION_MASK=<SYSGEN>

                DEFINE  EXE$M_NOSMPSANITY,-
                        VERSION_MASK=<SYSGEN>

                DEFINE  EXE$M_NOSPINWAIT,-
                        VERSION_MASK=<SYSGEN>

                DEFINE  BOO$A_SYSPARAM,-
                        VERSION_MASK=<SYSGEN>

                DEFINE  BOO$A_PRMBLK,-
                        VERSION_MASK=<SYSGEN>


                .ENDC   ; version
                .IF     NOT_DEFINED GETSYISW
                .IF     NOT_DEFINED VERSION
;
; OWN STORAGE:
;
                .IF     NDF,PRMSW               ;
                .PSECT  $$$$$SYSPARAM_DATA,PAGE,WRT,NOEXE,OVR
                .IFF                            ;
                .PSECT  $$$917A,PAGE
                .ENDC                           ;
                DEFINE  EXE$A_SYSPARAM          ; SYSTEM PARAMETER BASE
                .IF     DF,PRMSW                ;
BOO$A_SYSPARAM::                                ; BASE OF DEFAULTS
                .PSECT  $$$918,LONG             ; PARAMETER DESCRIPTOR LIST
BOO$A_PRMBLK::                                  ;
                .ENDC                           ;
                .IF     NDF,PRMSW               ;
                .PSECT  $$$$$SYSPARAM_DATA,PAGE,WRT,NOEXE,OVR
                .IFF
                .PSECT  $$$917A,PAGE
                .ENDC
                .ENDC   ; Version
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 27
X-101U18 SYSTEM TIME VARIABLES 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 1352          .SBTTL  SYSTEM TIME VARIABLES
 1353 ;
 1354 ;                 SYSTEM TIME VARIABLES
 1355 ;
 1356          DEFINE  EXE$GQ_TODCBASE,-       ; TIME OF DAY CLOCK BASE (BOOT TIME)
 1357                  VERSION_MASK=<SYSGEN>
 1358          .QUAD   ^X0091e2e116bc4000      ; 01-JAN-1989 00:00:00.00
 1359 ;
 1360          DEFINE  EXE$GL_TODR,-           ; TIME OF DAY REGISTER VALUE CORRESPONDING
 1361                  VERSION_MASK=<SYSGEN>
 1362          .LONG   1@28                    ; TO EXE$GQ_TODCBASE; IT'S NOT ZERO
 1363                                          ; BECAUSE ANYTHING LESS THAN 1@28
 1364                                          ; IS USED TO INDICATE CLOCK LOST POWER
 1365
 1366          .ENDC   ; NOT_DEFINED GETSYISW
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 28
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 1368          .SBTTL  SYSGEN PARAMETERS
 1369 ;
 1370 ;        SYSGEN PARAMETERS
 1371 ;
 1372
 1373 ; When the code in SYS$GETSYI/F$GETSYI/LIB$GETSYI invokes the SYI_GENERATE_TABLE
 1374 ; macro once, the PARAMETER macro (and therefore the SYI_ITEM_CODE macro)
 1375 ; will be invoked many times, generating the item-code control tables used
 1376 ; by the above peices of code.
 1377
 1378
 1379          ; Start the definition of the macro
 1380
 1381          .MACRO  SYI_GENERATE_TABLE
 1382
 1383          ; Define the GETSYI item-code which are Not SYSBOOT params
 1384
 1385          .IIF    DEFINED GETSYISW,       SYI_ITEMTABLES
 1386
 1387
 1388 ;
 1389 ; DEFAULT PAGE FAULT CLUSTER SIZE - SPECIFIES THE MAXIMUM NUMBER OF
 1390 ;       PAGES WHICH WILL BE READ FROM SECTIONS NOT SPECIFYING A CLUSTER FACTOR.
 1391 ;       THIS ALSO APPLIES TO PAGE FILE PAGES.
 1392 ;
 1393          PARAMETER       ADDRESS=SGN$GW_DFPFC,-  ;
 1394                          DEFAULT=32,-            ;
 1395                          MIN=0,-                 ;
 1396                          MAX=127,-               ;
 1397                          NAME=PFCDEFAULT,-       ;
 1398                          SIZE=WORD,-             ;
 1399                          TYPE=<DYNAMIC,SYS,MAJOR>,-      ;
 1400                          UNIT=Pages,-
 1401                          VERSION_MASK=<SYSGEN>
 1402 ;
 1403 ; DEFAULT PAGE TABLE PAGE FAULT CLUSTER SIZE - SPECIFIES THE MAXIMUM NUMBER OF
 1404 ;       OF PAGE TABLES TO ATTEMPT TO READ TO SATISFY A FAULT FOR A NON-RESIDENT
 1405 ;       PAGE TABLE.
 1406 ;
 1407          PARAMETER       ADDRESS=SGN$GB_PGTBPFC,-        ;
 1408                          DEFAULT=2,-             ;
 1409                          MIN=0,-                 ;
 1410                          MAX=127,-               ;
 1411                          NAME=PAGTBLPFC,-        ;
 1412                          SIZE=BYTE,-             ;
 1413                          TYPE=<DYNAMIC,SPECIAL>,-        ;
 1414                          UNIT=Pages,-
 1415                          VERSION_MASK=<SYSGEN>
 1416 ;
 1417 ;       PAGE FAULT CLUSTER FOR SYSTEM PAGING
 1418 ;
 1419          PARAMETER       ADDRESS=SGN$GB_SYSPFC,- ;
 1420                          DEFAULT=1,-             ;
 1421                          MIN=0,-                 ;
 1422                          MAX=127,-               ;
 1423                          NAME=SYSPFC,-           ;
 1424                          SIZE=BYTE,-             ;
```

```
                        TYPE=<SPECIAL>,-        ;
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>
;
; NUMBER OF KNOWN FILE LISTS - ESTABLISHES THE MAXIMUM NUMBER OF KNOWN
;       FILE LISTS THAT CAN BE MADE KNOWN TO THE SYSTEM.
;
        PARAMETER       ADDRESS=SGN$GB_KFILSTCT,-       ;
                        DEFAULT=4,-             ;
                        MIN=2,-                 ;
                        MAX=255,-               ;
                        NAME=KFILSTCNT,-        ;
                        SIZE=BYTE,-             ;
                        TYPE=<SYSGEN,SYS>,-     ;
                        UNIT=Slots,-
                        VERSION_MASK=<SYSGEN>

        .IF     NOT_DEFINED VERSION
        .IIF    NOT_DEFINED GETSYISW,   .ALIGN  WORD
        .ENDC
;
; GLOBAL SECTION COUNT - DETERMINES THE MAXIMUM NUMBER OF GLOBAL SECTIONS
;       WHICH CAN BE MADE KNOWN TO THE SYSTEM BY ALLOCATING THE NECESSARY
;       STORAGE FOR THE GST ENTRIES.
;

;       Note that 4095 global sections, coupled with the current size of a
;       section table entry, 8 longwords, allows all section table indices to
;       be represented as negative signed words.

        PARAMETER       ADDRESS=SGN$GW_GBLSECNT,-       ;
                        DEFAULT=250,-           ;
                        MIN=20,-                ;
                        MAX=4095,-              ;
                        NAME=GBLSECTIONS,-      ;
                        SIZE=WORD,-             ;
                        TYPE=<SYSGEN,SYS,MAJOR>,-       ;
                        UNIT=Sections,-
                        VERSION_MASK=<SYSGEN>
;
; GLOBAL PAGE COUNT - ESTABLISHES THE SIZE OF THE GLOBAL PAGE TABLE AND THE
;       LIMIT FOR THE TOTAL NUMBER OF GLOBAL PAGES THAT CAN BE CREATED.
;

        PARAMETER       ADDRESS=SGN$GL_MAXGPGCT,-       ;
                        DEFAULT=10000,-         ;
                        MIN=512,-               ;
                        NAME=GBLPAGES,-         ;
                        SIZE=LONG,-             ;
                        TYPE=<SYSGEN,SYS,MAJOR>,-               ;
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>
;
; GLOBAL PAGE PAGE FILE PAGE LIMIT - ESTABLISHES THE MAXIMUM NUMBER OF GLOBAL
; PAGES WITH PAGE FILE BACKING STORE THAT CAN BE CREATED.
;

```

```
1482          PARAMETER       ADDRESS=SGN$GL_GBLPAGFIL,-      ;
1483                          DEFAULT=1024,-          ;
1484                          MIN=128,-               ;
1485                          NAME=GBLPAGFIL,-                ;
1486                          SIZE=LONG,-             ;
1487                          TYPE=<SYS>,-            ;
1488                          UNIT=Pages,-
1489                          VERSION_MASK=<SYSGEN>
1490
1491 ;
1492 ; MAXIMUM PROCESS COUNT - DETERMINES THE MAXIMUM NUMBER OF PROCESSES
1493 ;
1494          PARAMETER       ADDRESS=SGN$GW_MAXPRCCT,        ;
1495                          DEFAULT=32,-            ;
1496                          MIN=12,-                ;
1497                          MAX=8192,-              ;
1498                          NAME=MAXPROCESSCNT,-    ;
1499                          SIZE=WORD,-             ;
1500                          TYPE=<SYSGEN,SYS,MAJOR>,-               ;
1501                          UNIT=Processes,-
1502                          VERSION_MASK=<SYSGEN>
1503
1504 ;
1505 ; PROCESS SCAN COUNT - DETERMINES THE MAXIMUM NUMBER OF PROCESSES TO SCAN
1506 ; FOR PRIORITY BOOSTING.
1507 ;
1508          PARAMETER       ADDRESS=SGN$GW_PIXSCAN,-;
1509                          DEFAULT=1,-             ;
1510                          MIN=0,-                 ;
1511                          MAX=8192,-              ;
1512                          NAME=PIXSCAN,-          ;
1513                          SIZE=WORD,-             ;
1514                          TYPE=<SPECIAL,DYNAMIC>,-;
1515                          UNIT=Processes,-
1516                          VERSION_MASK=<SYSGEN>
1517
1518
1519 ; NUMBER OF CPUS TO BOOT IN AN SMP SYSTEM, INCLUDING THE PRIMARY (BOOT)
1520 ; CPU AND ALL ATTAHED PROCESSORS
1521 ;
1522 ;
1523          PARAMETER       ADDRESS=SGN$GL_SMP_CPUS,-;
1524                          DEFAULT=-1,-            ;
1525                          MIN=0,-                 ;
1526                          NAME=SMP_CPUS,-         ;
1527                          SIZE=LONG,-             ;
1528                          TYPE=<MULTIPROCESSING,MAJOR>,-  ;
1529                          UNIT=<CPU bitmask>,-    ;
1530                          VERSION_MASK=<SYSGEN>
1531
1532          PARAMETER       ADDRESS=SGN$GL_SMP_CPUSH,-;
1533                          DEFAULT=0,-             ;
1534                          MIN=0,-                 ;
1535                          NAME=SMP_CPUSH,-        ;
1536                          SIZE=LONG,-             ;
1537                          TYPE=<SPECIAL>,-        ;
1538                          UNIT=<CPU bitmask>,-    ;
```

```
                        VERSION_MASK=<SYSGEN>

;
;       MULTIPROCESSING - Controls loading of spinlock code:
;               0: Never load (use UNI image)
;               1: Load MON image if MULTIPROCESSOR configuration else UNI image
;               2: Always load MON image
;               3: Load MIN image if MULTIPROCESSOR configuration else UNI image
;
        PARAMETER       ADDRESS=SGN$GB_MULTIPROCESSING, -       ;
                        DEFAULT=3,-     ;
                        MAX=3,-         ;
                        MIN=0,-         ;
                        NAME=MULTIPROCESSING,-  ;
                        SIZE=BYTE,-     ;
                        TYPE=<MULTIPROCESSING>,-        ;
                        UNIT=Coded-value,-
                        VERSION_MASK=<SYSGEN>
;
;       SMP_SANITY_CNT - Number of SMP sanity timer cycles until timeout.hardware cl
;
        PARAMETER       ADDRESS=SGN$GW_SMP_SANITY_CNT,- ;
;
; *** SMP NOTE ***
; The following is what we expect a reasonable default value to be. It
; has been increased by a factor of 10 to try to avoid premature timeouts
; while we gain experience with the timeout code.
;
;                       DEFAULT=30,-    ;
                        DEFAULT=300,-   ;
                        MAX=-1,-                ;
                        MIN=1,-         ;
                        NAME=SMP_SANITY_CNT,-   ;
                        SIZE=WORD,-     ;
                        TYPE=<MULTIPROCESSING>,-        ;
                        UNIT=<10ms.>,-
                        VERSION_MASK=<SYSGEN>
;
;       SMP_TICK_CNT - Number of clock ticks between SMP sanity timer cycles
;
        PARAMETER       ADDRESS=SGN$GW_SMP_TICK_CNT,- ;
;
; *** SMP NOTE ***
; The following is what we expect a reasonable default value to be. It
; has been increased by a factor of 10 to try to avoid premature timeouts
; while we gain experience with the timeout code.
;
;                       DEFAULT=3,-     ;
                        DEFAULT=30,-    ;
                        MAX=-1,-                ;
                        MIN=1,-         ;
                        NAME=SMP_TICK_CNT,-     ;
                        SIZE=WORD,-     ;
                        TYPE=<SPECIAL>,-        ;
                        UNIT=<10ms.>,-
                        VERSION_MASK=<SYSGEN>
;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 32
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 1596 ;       SMP_SPINWAIT - Normal SMP busywait timeout interval
 1597 ;
 1598         PARAMETER       ADDRESS=SGN$GL_SMP_SPINWAIT,- ;
 1599 ;
 1600 ; *** SMP NOTE ***
 1601 ; The following is what we expect a reasonable default value to be. It
 1602 ; has been increased by a factor of 10 to try to avoid premature timeouts
 1603 ; while we gain experience with the timeout code.
 1604 ;
 1605 ;                       DEFAULT=10000,- ; 100 ms.
 1606                         DEFAULT=100000,-        ;
 1607                         MAX=8388607,-   ; set to prevent overflow on normalization
 1608                         MIN=1,-         ;
 1609                         NAME=SMP_SPINWAIT,-     ;
 1610                         SIZE=LONG,-     ;
 1611                         TYPE=<MULTIPROCESSING>,-        ;
 1612                         UNIT=<10 usec.>,-
 1613                         VERSION_MASK=<SYSGEN>
 1614 ;
 1615 ; SMP_LNGSPINWAIT - LONG SMP busywait timeout interval
 1616 ;
 1617         PARAMETER       ADDRESS=SGN$GL_SMP_LNGSPINWAIT,- ;
 1618 ;
 1619 ; *** SMP NOTE ***
 1620 ; The following is what we expect a reasonable default value to be. It
 1621 ; has been increased by a factor of 10 to try to avoid premature timeouts
 1622 ; while we gain experience with the timeout code.
 1623 ;
 1624 ;                       DEFAULT=300000,-        ; 3 seconds
 1625                         DEFAULT=3000000,-       ;
 1626                         MAX=8388607,-   ; set to prevent overflow on normalization
 1627                         MIN=1,-         ;
 1628                         NAME=SMP_LNGSPINWAIT,-  ;
 1629                         SIZE=LONG,-     ;
 1630                         TYPE=<MULTIPROCESSING>,-        ;
 1631                         UNIT=<10 usec.>,-
 1632                         VERSION_MASK=<SYSGEN>
 1633
 1634 ; PROCESS SECTION COUNT - GUARANTEED NUMBER OF PROCESS SECTIONS THAT CAN
 1635 ;       BE CREATED.  DEPENDING ON SIZE OF WORKING SET, THE ACTUAL NUMBER
 1636 ;       OF SECTIONS CAN ACTUALLY BE GREATER.
 1637 ;
 1638         PARAMETER       ADDRESS=SGN$GW_MAXPSTCT,-       ;
 1639                         DEFAULT=32,-            ;
 1640                         MIN=5,-                 ;
 1641                         MAX=1024,-              ;
 1642                         NAME=PROCSECTCNT,-      ;
 1643                         SIZE=WORD,-             ;
 1644                         TYPE=<SYSGEN,SYS>,-             ;
 1645                         UNIT=Sections,-
 1646                         VERSION_MASK=<SYSGEN>
 1647
 1648
 1649
 1650 ;
 1651 ;
 1652         PARAMETER       ADDRESS=SGN$GL_MINWSCNT,-       ;
```

```
1653                            DEFAULT=20,-            ;
1654                            MIN=10,-                ;
1655                            NAME=MINWSCNT,-         ;
1656                            SIZE=LONG,-             ;
1657                            TYPE=<STATIC,SYSGEN,SYS>,-      ;
1658                            UNIT=Pages,-
1659                            VERSION_MASK=<SYSGEN>
1660
1661
1662 ;
1663 ; NUMBER OF PAGING FILES - DETERMINES THE MAXIMUM NUMBER OF PAGING FILES
1664 ;       THAT CAN BE MADE KNOW TO THE SYSTEM.
1665 ;
1666            PARAMETER       ADDRESS=SGN$GW_PAGFILCT,-       ;
1667                            DEFAULT=2,-             ;
1668                            MIN=1,-                 ;
1669                            MAX=63,-                ;
1670                            NAME=PAGFILCNT,-        ;
1671                            SIZE=WORD,-             ;
1672                            TYPE=<SYS,SYSGEN>,-     ;
1673                            UNIT=Files,-
1674                            VERSION_MASK=<SYSGEN>
1675
1676 ;
1677 ; NUMBER OF SWAP FILES - ESTABLISHES THE MAXIMUM NUMBER OF SWAPFILES THAT
1678 ;       CAN BE MADE KNOWN TO THE SYSTEM.
1679 ;
1680            PARAMETER       ADDRESS=SGN$GW_SWPFILES,-       ;
1681                            DEFAULT=2,-             ;
1682                            MIN=0,-                 ;
1683                            MAX=63,-                ;
1684                            NAME=SWPFILCNT,-        ;
1685                            SIZE=WORD,-             ;
1686                            TYPE=<SYS,SYSGEN>,-     ;
1687                            UNIT=Files,-
1688                            VERSION_MASK=<SYSGEN>
1689
1690
1691 ;
1692 ;
1693            PARAMETER       ADDRESS=SGN$GL_SYSDWSCT,-       ;
1694                            DEFAULT=500,-           ;
1695                            MIN=40,-                ;
1696                            MAX=16384,-             ;
1697                            NAME=SYSMWCNT,-         ;
1698                            SIZE=LONG,-             ;
1699                            TYPE=<SYSGEN,SYS,MAJOR>,-       ;
1700                            UNIT=Pages,-
1701                            VERSION_MASK=<SYSGEN>
1702
1703 ;
1704 ; INTERRUPT STACK SIZE - ESTABLISHES THE SIZE OF THE INTERRUPT STACK IN PAGES
1705 ;
1706            PARAMETER       ADDRESS=SGN$GW_ISPPGCT,-        ;
1707                            DEFAULT=4,-             ;
1708                            MIN=1,-                 ;
1709                            NAME=INTSTKPAGES,-              ;
```

```
                        SIZE=WORD,-                     ;
                        TYPE=<SYS,SYSGEN>,-             ;
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>
;
; AMOUNT OF EXTRA INTERRUPT STACK TO LEAVE WHEN DOING DEADLOCK SEARCH
;
        PARAMETER       ADDRESS=LCK$GL_EXTRASTK,-       ;
                        DEFAULT=512,-                   ;
                        MIN=256,-                       ;
                        NAME=DLCKEXTRASTK,-             ;
                        SIZE=LONG,-                     ;
                        TYPE=<SPECIAL>,-                ;
                        UNIT=Bytes,-                    ;
                        VERSION_MASK=<SYSGEN>

;
; BALANCE SET COUNT - DETERMINES THE MAXIMUM NUMBER OF PROCESS THAT CAN BE
;       BE CONCURRENTLY RESIDENT.
;
        PARAMETER       ADDRESS=SGN$GL_BALSETCT,-       ;
                        DEFAULT=16,-    ;
                        MIN=4,-         ;
                        MAX=1024,-      ;
                        NAME=BALSETCNT,-        ;
                        SIZE=LONG,-     ;
                        TYPE=<SYSGEN,SYS,MAJOR>,-       ;
                        UNIT=Slots,-
                        VERSION_MASK=<SYSGEN>
;
; COUNT OF PRE-ALLOCATED I/O PACKETS - DETERMINES THE NUMBER OF I/O PACKETS
;       TO BE PRE-ALLOCATED AND LINKED TOGETHER FOR FAST ALLOCATION AND
;       DEALLOCATION.
;
        PARAMETER       ADDRESS=SGN$GL_IRPCNT,- ;
                        DEFAULT=60,-            ;
                        MIN=0,-                 ;
                        MAX=32768,-             ;
                        NAME=IRPCOUNT,-         ;
                        SIZE=LONG,-             ;
                        TYPE=<SYSGEN,MAJOR,SYS>,-       ;
                        UNIT=Packets,-
                        VERSION_MASK=<SYSGEN>
;
; NUMBER OF PACKETS TO WHICH THE IRPLIST MAY BE EXTENDED.
;
        PARAMETER       ADDRESS=SGN$GL_IRPCNTV,-;
                        DEFAULT=250,-           ;
                        MIN=0,-                 ;
                        MAX=32768,-             ;
                        NAME=IRPCOUNTV,-        ;
                        SIZE=LONG,-             ;
                        TYPE=<SYSGEN,SYS>,-     ;
                        UNIT=Packets,-
                        VERSION_MASK=<SYSGEN>

;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 35
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 1767 ; MAXIMUM SIZE OF PROCESS WORKING SET. DETERMINES THE SYSTEM WIDE MAXIMUM
 1768 ;       SIZE OF A PROCESS WORKING SET REGARDLESS OF PROCESS QUOTA.
 1769 ;
 1770         PARAMETER       ADDRESS=SGN$GL_MAXWSCNT,-       ;
 1771                         DEFAULT=1024,-          ;
 1772                         MIN=60,-                ;
 1773                         MAX=100000,-            ;
 1774                         NAME=WSMAX,-            ;
 1775                         SIZE=LONG,-             ;
 1776                         TYPE=<SYSGEN,SYS,MAJOR>,-       ;
 1777                         UNIT=Pages,-
 1778                         VERSION_MASK=<SYSGEN>
 1779
 1780 ;
 1781 ; NON-PAGED DYNAMIC POOL - DETERMINES THE NUMBER OF BYTES TO ALLOCATE FOR
 1782 ;       THE NON-PAGED DYNAMIC POOL.
 1783 ;
 1784         PARAMETER       ADDRESS=SGN$GL_NPAGEDYN,-       ;
 1785                         DEFAULT=300032,-        ; 586 pages
 1786                         MIN=16384,-             ;
 1787                         NAME=NPAGEDYN,-         ;
 1788                         SIZE=LONG,-             ;
 1789                         TYPE=<SYSGEN,SYS,MAJOR>,-       ;
 1790                         UNIT=Bytes,-
 1791                         VERSION_MASK=<SYSGEN>
 1792 ;
 1793 ; NON-PAGED DYNAMIC POOL - DETERMINES THE NUMBER OF BYTES TO WHICH
 1794 ;       THE NON-PAGED DYNAMIC POOL MAY BE EXTENDED.  THIS PARAMETER
 1795 ;       IS USED TO ALLOCATE THE NECESSARY PAGE TABLE ENTRIES.
 1796 ;
 1797         PARAMETER       ADDRESS=SGN$GL_NPAGEVIR,-       ;
 1798                         DEFAULT=1000000,-               ;
 1799                         MIN=16384,-             ;
 1800                         NAME=NPAGEVIR,-         ;
 1801                         SIZE=LONG,-             ;
 1802                         TYPE=<SYSGEN,SYS>,-     ;
 1803                         UNIT=Bytes,-
 1804                         VERSION_MASK=<SYSGEN>
 1805 ;
 1806 ; PAGED DYNAMIC POOL - DETERMINES THE NUMBER OF BYTES TO ALLOCATE FOR THE
 1807 ;       PAGED DYNAMIC POOL.
 1808 ;
 1809         PARAMETER       ADDRESS=SGN$GL_PAGEDYN,-        ;
 1810                         DEFAULT=190000,-        ;
 1811                         MIN=10240,-             ;
 1812                         NAME=PAGEDYN,-          ;
 1813                         SIZE=LONG,-             ;
 1814                         TYPE=<SYSGEN,SYS,MAJOR>,-               ;
 1815                         UNIT=Bytes,-
 1816                         VERSION_MASK=<SYSGEN>
 1817
 1818 ;
 1819 ; MAXIMUM VIRTUAL PAGE COUNT - DETERMINES THE TOTAL NUMBER OF PAGES THAT
 1820 ;       CAN BE MAPPED FOR A PROCESS, WHICH CAN BE DIVIDED IN ANY FASHION
 1821 ;       BETWEEN P0 AND P1 SPACE.
 1822 ;
 1823         PARAMETER       ADDRESS=SGN$GL_MAXVPGCT,-       ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 36
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 1824                           DEFAULT=8192,-          ;
 1825                           MIN=512,-               ;
 1826                           MAX=600000,-            ;
 1827                           NAME=VIRTUALPAGECNT,-   ;
 1828                           SIZE=LONG,-             ;
 1829                           TYPE=<SYSGEN,SYS,MAJOR>,-       ;
 1830                           UNIT=Pages,-
 1831                           VERSION_MASK=<SYSGEN>
 1832
 1833 ;
 1834 ; REQUESTED SPT EXTENSION - NUMBER OF ADDITIONAL SPT SLOT TO ALLOW
 1835 ;
 1836           PARAMETER       ADDRESS=SGN$GL_SPTREQ,- ;
 1837                           DEFAULT=2500,-          ;
 1838                           NAME=SPTREQ,-           ;
 1839                           SIZE=LONG,-             ;
 1840                           TYPE=<SYS,SYSGEN>,-     ;
 1841                           UNIT=Pages,-
 1842                           VERSION_MASK=<SYSGEN>
 1843 ;
 1844 ; EXTRA USER STACK AUTOMATICALLY PROVIDED BY THE IMAGE ACTIVATOR
 1845 ;       SO THAT THE OPERATING SYSTEM CAN RECOVER FROM A STACK OVERFLOW.
 1846 ;
 1847           PARAMETER       ADDRESS=SGN$GL_EXUSRSTK,- ;
 1848                           DEFAULT=<2*512>,-       ;
 1849                           MIN=<2*512>,-           ;
 1850                           NAME=EXUSRSTK,-         ;
 1851                           SIZE=LONG,-             ;
 1852                           TYPE=<SPECIAL>,-        ;
 1853                           UNIT=<Pages*512>,-      ;
 1854                           VERSION_MASK=<SYSGEN>
 1855 ;
 1856 ; NUMBER OF LARGE REQUEST PACKETS TO ALLOCATE TO THE LRP LOOK ASIDE LIST
 1857 ;
 1858           PARAMETER       ADDRESS=SGN$GL_LRPCNT,- ;
 1859                           DEFAULT=4,-             ;
 1860                           MIN=0,-                 ;
 1861                           MAX=4096,-              ;
 1862                           NAME=LRPCOUNT,-         ;
 1863                           SIZE=LONG,-             ;
 1864                           TYPE=<SYS,SYSGEN,MAJOR>,-       ;
 1865                           UNIT=<Packets>,-                ;
 1866                           VERSION_MASK=<SYSGEN>
 1867 ;
 1868 ; NUMBER OF LARGE REQUEST PACKETS TO WHICH THE LRP LOOK ASIDE LIST
 1869 ; MAY BE EXTENDED.  USED TO ALLOCATE THE APPROPRIATE VIRTUAL SPACE.
 1870 ;
 1871           PARAMETER       ADDRESS=SGN$GL_LRPCNTV,-;
 1872                           DEFAULT=20,-            ;
 1873                           MIN=0,-                 ;
 1874                           MAX=4096,-              ;
 1875                           NAME=LRPCOUNTV,-                ;
 1876                           SIZE=LONG,-             ;
 1877                           TYPE=<SYS,SYSGEN>,-     ;
 1878                           UNIT=<Packets>,-        ;
 1879                           VERSION_MASK=<SYSGEN>
 1880 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 37
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 1881 ; SIZE OF LARGE REQUEST PACKETS (BYTES)
 1882 ;
 1883         PARAMETER       ADDRESS=SGN$GL_LRPSIZE,-;
 1884                         DEFAULT=1504,-          ; Assume Ethernet
 1885                         MIN=256,-               ;
 1886                         MAX=16384,-             ;
 1887                         NAME=LRPSIZE,-          ;
 1888                         SIZE=LONG,-             ;
 1889                         TYPE=<SYS,SYSGEN>,-     ;
 1890                         UNIT=<Bytes>,-          ;
 1891                         VERSION_MASK=<SYSGEN>
 1892 ;
 1893 ; MINIMUM ALLOCATION REQUEST FOR LARGE REQUEST PACKETS (BYTES)
 1894 ;
 1895         PARAMETER       ADDRESS=SGN$GL_LRPMIN,-;
 1896                         DEFAULT=1088,-          ;
 1897                         MIN=256,-               ;
 1898                         MAX=16384,-             ;
 1899                         NAME=LRPMIN,-           ;
 1900                         SIZE=LONG,-             ;
 1901                         TYPE=<SPECIAL>,-        ;
 1902                         UNIT=<Bytes>,-          ;
 1903                         VERSION_MASK=<SYSGEN>
 1904 ;
 1905 ; NUMBER OF SMALL REQUEST PACKETS TO ALLOCATE TO THE SRP LOOK ASIDE LIST
 1906 ;
 1907         PARAMETER       ADDRESS=SGN$GL_SRPCNT,-;
 1908                         DEFAULT=120,-           ;
 1909                         MIN=0,-                 ;
 1910                         MAX=131072,-            ;
 1911                         NAME=SRPCOUNT,-         ;
 1912                         SIZE=LONG,-             ;
 1913                         TYPE=<SYS,SYSGEN,MAJOR>,-       ;
 1914                         UNIT=<Packets>,-        ;
 1915                         VERSION_MASK=<SYSGEN>
 1916 ;
 1917 ; NUMBER OF SMALL REQUEST PACKETS TO WHICH THE SRP LOOK ASIDE LIST
 1918 ; MAY BE EXTENDED.  USED TO ALLOCATE THE APPROPRIATE VIRTUAL SPACE.
 1919 ;
 1920         PARAMETER       ADDRESS=SGN$GL_SRPCNTV,-;
 1921                         DEFAULT=250,-           ;
 1922                         MIN=0,-                 ;
 1923                         MAX=131072,-            ;
 1924                         NAME=SRPCOUNTV,-        ;
 1925                         SIZE=LONG,-             ;
 1926                         TYPE=<SYS,SYSGEN>,-     ;
 1927                         UNIT=<Packets>,-        ;
 1928                         VERSION_MASK=<SYSGEN>
 1929 ;
 1930 ; SIZE OF SMALL REQUEST PACKETS (BYTES)
 1931 ;
 1932         PARAMETER       ADDRESS=SGN$GL_SRPSIZE,-;
 1933                         DEFAULT=96,-            ;
 1934                         MIN=96,-                ;
 1935                         MAX=144,-               ;
 1936                         NAME=SRPSIZE,-          ;
 1937                         SIZE=LONG,-             ;
```

```
1938                            TYPE=<SPECIAL>,-        ;
1939                            UNIT=<Bytes>,-          ;
1940                            VERSION_MASK=<SYSGEN>
1941 ;
1942 ; MINIMUM ALLOCATION REQUEST FOR SMALL REQUEST PACKETS (BYTES)
1943 ;
1944            PARAMETER        ADDRESS=SGN$GL_SRPMIN,-;
1945                            DEFAULT=32,-            ;
1946                            MIN=0,-                 ;
1947                            MAX=144,-               ;
1948                            NAME=SRPMIN,-           ;
1949                            SIZE=LONG,-             ;
1950                            TYPE=<SPECIAL>,-        ;
1951                            UNIT=<Bytes>,-          ;
1952                            VERSION_MASK=<SYSGEN>
1953 ;
1954 ;       PERMANENT I/O CHANNEL COUNT - SPECIFES THE NUMBER OF PERMANENT I/O
1955 ;       CHANNELS TO PROVIDE.
1956 ;
1957            PARAMETER        ADDRESS=SGN$GW_PCHANCNT,-       ;
1958                            DEFAULT=127,-           ;
1959                            MIN=31,-                ;
1960                            MAX=2047,-              ;
1961                            NAME=CHANNELCNT,-       ;
1962                            SIZE=WORD,-             ;
1963                            TYPE=<SPECIAL>,-        ;
1964                            UNIT=Channels,-
1965                            VERSION_MASK=<SYSGEN>
1966 ;
1967 ;       PROCESS I/O PAGES - SPECIFIES THE NUMBER OF PAGES OF PROCESS
1968 ;       I/O ADDRESS SPACE FOR PRCSTRT TO CREATE.
1969 ;
1970            PARAMETER        ADDRESS=SGN$GW_PIOPAGES, - ;
1971                            DEFAULT=245, -          ;
1972                            MIN=10, -               ;
1973                            NAME=PIOPAGES, -        ;
1974                            SIZE=WORD, -            ;
1975                            TYPE=<SPECIAL>, -       ;
1976                            UNIT=Pages,-
1977                            VERSION_MASK=<SYSGEN>
1978 ;
1979 ;       CONTROL REGION IMPURE PAGES - SPECIFIES THE NUMBER OF PAGES OF
1980 ;       PROCESS ALLOCATION REGION SPACE FOR PRCSTRT TO CREATE.
1981 ;
1982            PARAMETER        ADDRESS=SGN$GW_CTLPAGES, - ;
1983                            DEFAULT=50, -           ;
1984                            MIN=10, -               ;
1985                            NAME=CTLPAGES, -        ;
1986                            SIZE=WORD, -            ;
1987                            TYPE=<SPECIAL>, -       ;
1988                            UNIT=Pages,-
1989                            VERSION_MASK=<SYSGEN>
1990
1991 ;
1992 ; LIMIT ON USE OF THE PROCESS ALLOCATION REGION BY IMAGE REQUESTS
1993 ;
1994            PARAMETER        ADDRESS=SGN$GW_CTLIMGLIM, - ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 39
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 1995                           DEFAULT=35,-            ;
 1996                           MIN=0, -                ;
 1997                           NAME=CTLIMGLIM, -       ;
 1998                           SIZE=WORD, -            ;
 1999                           TYPE=<SPECIAL>, -       ;
 2000                           UNIT=Pages,-
 2001                           VERSION_MASK=<SYSGEN>
 2002 ;
 2003 ; DEFAULT NUMBER OF PAGES OF IMAGE I/O ADDRESS SPACE USED BY
 2004 ;       THE IMAGE ACTIVATOR IF NOT SPECIFIED AT PROGRAM LINK TIME.
 2005 ;
 2006         PARAMETER         ADDRESS=SGN$GW_IMGIOCNT,-       ;
 2007                           DEFAULT=64,-            ;
 2008                           MIN=32,-                ;
 2009                           NAME=IMGIOCNT,-         ;
 2010                           SIZE=WORD,-             ;
 2011                           TYPE=<DYNAMIC,SPECIAL>,-        ;
 2012                           UNIT=Pages,-
 2013                           VERSION_MASK=<SYSGEN>
 2014         .PAGE
 2015         .SBTTL  CONTROL PARAMETERS
 2016
 2017         .IF     NOT_DEFINED VERSION
 2018         .IIF    NOT_DEFINED GETSYISW,   .ALIGN  WORD
 2019         .ENDC
 2020
 2021 ;
 2022 ;       GENERAL SYSTEM CONTROL PARAMETERS
 2023 ;
 2024         PARAMETER         ADDRESS=SCH$GW_QUAN,-   ; PROCESS QUANTUM
 2025                           DEFAULT=-20,-           ; NEGATED
 2026                           MIN=2,-                 ;
 2027                           MAX=32767,-             ;
 2028                           NAME=QUANTUM,-          ;
 2029                           SIZE=WORD,-             ;
 2030                           TYPE=<DYNAMIC,SYS,NEG,MAJOR>,-  ;
 2031                           UNIT=10Ms,-             ;
 2032                           VERSION_MASK=<SYSGEN>
 2033 ;
 2034 ;       MODIFIED PAGE WRITER CONTROL PARAMETERS
 2035 ;
 2036         DEFINE  MPW$AW_INITVAL,-
 2037                 VERSION_MASK=<SYSGEN>
 2038
 2039 ;
 2040 ;       PAGE WRITE CLUSTER FACTOR - SPECIFIES THE NUMBER OF PAGES TO ATTEMPT
 2041 ;       TO WRITE AS A SINGLE I/O TRANSFER TO CONTIGUOUS DISK.
 2042 ;
 2043         PARAMETER         ADDRESS=MPW$GW_MPWPFC,- ;
 2044                           DEFAULT=120,-           ;
 2045                           MIN=16,-                ;
 2046                           MAX=120,-               ;
 2047                           NAME=MPW_WRTCLUSTER,-   ;
 2048                           SIZE=WORD,-             ;
 2049                           TYPE=<SYSGEN,SYS>,-     ;
 2050                           UNIT=Pages,-
 2051                           VERSION_MASK=<SYSGEN>
```

```
2052 ;
2053 ;       MODIFIED PAGE LIST HIGH LIMIT - THRESHOLD AT WHICH TO BEGIN WRITING
2054 ;       MODIFIED PAGES.
2055 ;
2056         PARAMETER       ADDRESS=MPW$GW_HILIM,-  ;
2057                         DEFAULT=500,-           ;
2058                         MIN=0,-                 ;
2059                         MAX=65535,-             ;
2060                         NAME=MPW_HILIMIT,-      ;
2061                         SIZE=WORD,-             ;
2062                         TYPE=<SYSGEN,SYS>,-     ;
2063                         UNIT=Pages,-
2064                         VERSION_MASK=<SYSGEN>
2065 ;
2066 ;       MODIFIED PAGE LIST LOW LIMIT - THRESHOLD AT WHICH MODIFIED PAGE WRITING
2067 ;       WILL NORMALLY STOP.  WRITING STARTED AT THE HIGH LIMIT AND PAGES
2068 ;       ARE WRITTEN IN CHUNKS CONTROLLED BY THE CLUSTER FACTOR. WHEN THE
2069 ;       LENGTH OF THE MODIFIED PAGE LIST HAS BEEN REDUCED BELOW THE LOW LIMIT,
2070 ;       WRITING CEASES UNTIL ENOUGH PAGES HAVE BEEN ADDED TO EXCEED THE
2071 ;       HIGH LIMIT.
2072 ;
2073         PARAMETER       ADDRESS=MPW$GW_LOLIM,-  ;
2074                         DEFAULT=32,-            ;
2075                         MIN=0,-                 ;
2076                         MAX=65535,-             ;
2077                         NAME=MPW_LOLIMIT,-      ;
2078                         SIZE=WORD,-             ;
2079                         TYPE=<SYSGEN,SYS>,-     ;
2080                         UNIT=Pages,-
2081                         VERSION_MASK=<SYSGEN>
2082 ;
2083 ; MODIFIED PAGE WRITER I/O LIMIT. THIS PARAMETER CONTROLS THE MAXIMUM NUMBER OF
2084 ; CONCURRENT I/O TRANSFERS INITIATED BY THE MODIFIED PAGE WRITER.
2085 ;
2086         PARAMETER       ADDRESS=MPW$GB_IOLIM,-  ;
2087                         DEFAULT=4,-             ;
2088                         MIN=1,-                 ;
2089                         MAX=127,-               ;
2090                         NAME=MPW_IOLIMIT,-      ;
2091                         SIZE=BYTE,-             ;
2092                         TYPE=<SYSGEN,SYS>,-     ;
2093                         UNIT=I/O,-
2094                         VERSION_MASK=<SYSGEN>
2095 ;
2096 ; MODIFIED PAGE WRITER I/O PRIORITY.  THIS PARAMETER SETS THE PRIORITY OF
2097 ; I/O TRANSFERS INITIATED BY THE MODIFIED PAGE WRITER.
2098 ;
2099         PARAMETER       ADDRESS=MPW$GB_PRIO,-   ;
2100                         DEFAULT=4,-             ;
2101                         MIN=0,-                 ;
2102                         MAX=31,-                ;
2103                         NAME=MPW_PRIO,-         ;
2104                         SIZE=BYTE,-             ;
2105                         TYPE=<SPECIAL,DYNAMIC>,-        ;
2106                         VERSION_MASK=<SYSGEN>
2107 ;
2108 ; SWAPPER I/O PRIORITY.  THIS PARAMETER SETS THE PRIORITY OF
```

```
2109 ; I/O TRANSFERS INITIATED BY THE SWAPPER.
2110 ;
2111         PARAMETER       ADDRESS=SWP$GB_PRIO,-   ;
2112                         DEFAULT=4,-             ;
2113                         MIN=0,-                 ;
2114                         MAX=31,-                ;
2115                         NAME=SWP_PRIO,- ;
2116                         SIZE=BYTE,-             ;
2117                         TYPE=<SPECIAL,DYNAMIC>,-
2118                         VERSION_MASK=<SYSGEN>
2119 ;
2120 ; MODIFIED PAGE WRITER LOWER LIMIT THRESHOLD STOPPING USE OF MODIFIED PAGE
2121 ; WRITER FROM BEING USED AS PRIMARY MECHANISM TO RECOVER MEMORY.
2122 ;
2123         PARAMETER       ADDRESS=MPW$GL_THRESH,- ;
2124                         DEFAULT=200,-           ;
2125                         MIN=0,-                 ;
2126                         MAX=65535,-             ;
2127                         NAME=MPW_THRESH,-       ;
2128                         SIZE=LONG,-             ;
2129                         TYPE=<SYS,DYNAMIC>,-
2130                         UNIT = Pages,-
2131                         VERSION_MASK=<SYSGEN>
2132 ;
2133 ; MODIFIED PAGE WRITER BUSY WAIT LIMIT.  THIS IS USED AS A THRESHOLD OF
2134 ; WHEN TO PUT A PROCESS INTO RESOURCE WAIT IF IT IS GENERATING A MODIFIED
2135 ; PAGE AND THE SIZE OF THE MODIFIED LIST IS GREATER THAN THIS PARAMETER.
2136 ;
2137         PARAMETER       ADDRESS=MPW$GL_WAITLIM,-;
2138                         DEFAULT=620,-           ;
2139                         MIN=0,-                 ;
2140                         MAX=65535,-             ;
2141                         NAME=MPW_WAITLIMIT,-    ;
2142                         SIZE=LONG,-             ;
2143                         TYPE=<SYS,DYNAMIC>,-
2144                         UNIT= Pages,-
2145                         VERSION_MASK=<SYSGEN>
2146 ;
2147 ; MODIFIED PAGE WRITER BUSY WAIT LOW LIMIT.  THIS IS USED AS A THRESHOLD OF
2148 ; WHEN TO DECLARE A RESOURCE AVAILABLE EVENT TO RESUME PROCESSES THAT WERE PUT
2149 ; INTO THE MPW BUSY RESOURCE WAIT STATE.
2150 ;
2151         PARAMETER       ADDRESS=MPW$GL_LOWAITLIM,-;
2152                         DEFAULT=380,-           ;
2153                         MIN=0,-                 ;
2154                         MAX=65535,-             ;
2155                         NAME=MPW_LOWAITLIMIT,-;
2156                         SIZE=LONG,-             ;
2157                         UNIT= Pages,-
2158                         TYPE=<SYS,DYNAMIC>      ;
2159 ;
2160 ; MAXIMUM NUMBER OF WORKING SET LIST ENTRIES THAT MAY BE SKIPPED WHILE
2161 ; SCANNING FOR A GOOD ENTRY TO DISCARD.  SET TO 0 TO DISABLE SKIPPING.
2162 ;
2163         PARAMETER       ADDRESS=SGN$GW_WSLMXSKP,-       ;
2164                         DEFAULT=8,-             ;
2165                         MIN=0,-                 ;
```

```
                        MAX=512,-               ;
                        NAME=TBSKIPWSL,-                ;
                        SIZE=WORD,-             ;
                        TYPE=<DYNAMIC,SPECIAL>,-        ;
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>

;
; Maximum number of physical pages to be used - permits testing of smaller
; memory configurations without actually removing memory boards.
;
        PARAMETER       ADDRESS=MMG$GL_PHYPGCNT,-       ;
                        DEFAULT=1047552,-       ;
                        MIN=2048,-      ;
                        MAX=1047552,-   ;
                        NAME=PHYSICALPAGES,-    ;
                        SIZE=LONG,-     ;
                        TYPE=<SPECIAL>,-        ;
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>
;
; Page fault rate lower threshold.  This parameter sets the lower page fault rate
; threshold for automatic working set size adjustment.
;
        PARAMETER       ADDRESS=SCH$GL_PFRATL,- ;
                        DEFAULT=0,-     ;
                        MIN=0,- ;
                        NAME=PFRATL,-   ;
                        SIZE=LONG,-     ;
                        TYPE=<SYS,DYNAMIC,MAJOR>,-      ;
                        UNIT=Flts/10Sec,-       ;
                        VERSION_MASK=<SYSGEN>
;
; Page fault rate high threshold.  This parameter sets the upper page fault
; rate threshold for automatic working set adjustment.
;
        PARAMETER       ADDRESS=SCH$GL_PFRATH,- ;
                        DEFAULT=120,-   ;
                        MIN=0,- ;
                        NAME=PFRATH,-   ;
                        SIZE=LONG,-     ;
                        TYPE=<SYS,DYNAMIC,MAJOR>,-      ;
                        UNIT=Flts/10Sec,-
                        VERSION_MASK=<SYSGEN>
;
; Page fault rate system threshold.  This parameter sets the target system page
; fault threshold.
;
        PARAMETER       ADDRESS=SCH$GL_PFRATS,- ;
                        DEFAULT=0,-     ;
                        MIN=0,- ;
                        NAME=PFRATS,-   ;
                        SIZE=LONG,-     ;
                        TYPE=<SPECIAL,DYNAMIC>,-        ;
                        UNIT=Flts/10Sec,-
                        VERSION_MASK=<SYSGEN>
;
```

```
2223 ; Working set increment.  This parameter sets the number of pages to increase the
2224 ; working set size to compensate for a high page fault rate.
2225 ;
2226         PARAMETER       ADDRESS=SCH$GL_WSINC,-  ;
2227                         DEFAULT=150,-   ;
2228                         MIN=0,- ;
2229                         NAME=WSINC,-    ;
2230                         SIZE=LONG,-     ;
2231                         TYPE=<SYS,DYNAMIC,MAJOR>,-      ;
2232                         UNIT=Pages,-
2233                         VERSION_MASK=<SYSGEN>
2234 ;
2235 ; Working set decrement.  This parameter sets the number of pages to decrease
2236 ; the working set to compensate for a page fault rate below the lower threshold,
2237 ; or alternatively (if PFRATL=0), when reclaiming memory in quantum end
2238 ; processing from a dormant, COMputable process during memory-tight periods.
2239 ;
2240 ; The default is chosen to be relatively prime to WSINC, and approximately
2241 ; equal to (default BORROWLIM) - (default GROWLIM).
2242 ;
2243         PARAMETER       ADDRESS=SCH$GL_WSDEC,-  ;
2244                         DEFAULT=250,-   ;
2245                         MIN=0,- ;
2246                         NAME=WSDEC,-    ;
2247                         SIZE=LONG,-     ;
2248                         TYPE=<SYS,DYNAMIC,MAJOR>,-      ;
2249                         UNIT=Pages,-
2250                         VERSION_MASK=<SYSGEN>
2251 ;
2252 ;
2253         PARAMETER       ADDRESS=SCH$GL_AWSMIN,- ;
2254                         DEFAULT=50,-    ;
2255                         MIN=0,-         ;
2256                         NAME=AWSMIN,-   ;
2257                         SIZE=LONG,-     ;
2258                         TYPE=<SYS,DYNAMIC>,-;
2259                         UNIT=Pages,-
2260                         VERSION_MASK=<SYSGEN>
2261 ;
2262 ; Working set measurement interval.  Sets the minimum interval of compute
2263 ; time for the measurement of page fault rate.
2264 ;
2265         PARAMETER       ADDRESS=SCH$GL_AWSTIME,-        ;
2266                         DEFAULT=20,-    ;
2267                         MIN=1,-         ;
2268                         NAME=AWSTIME,-  ;
2269                         SIZE=LONG,-     ;
2270                         TYPE=<SYS,DYNAMIC>,-;
2271                         UNIT=10Ms,-
2272                         VERSION_MASK=<SYSGEN>
2273 ;
2274 ; Swap rate control.  This parameter sets the swapping rate and serves to limit
2275 ; the consumption of disk bandwidth by swapping.
2276 ;
2277         PARAMETER       ADDRESS=SCH$GL_SWPRATE,-        ;
2278                         DEFAULT=500,-   ;
2279                         MIN=0,- ;
```

```
                        NAME=SWPRATE,-  ;
                        SIZE=LONG,-     ;
                        TYPE=<SPECIAL,DYNAMIC>,-        ;
                        UNIT=10Ms/Swap,-
                        VERSION_MASK=<SYSGEN>
;
; Desired process page count for an outswap swap.  This parameter sets the
; number of pages to attempt to reduce a working set to before starting the
; outswap.
;
        PARAMETER       ADDRESS=SWP$GL_SWPPGCNT,-
                        DEFAULT=288,-
                        MIN=0,-
                        NAME=SWPOUTPGCNT,-
                        SIZE=LONG,-
                        TYPE=<SYS,DYNAMIC>,-
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>
;
; Swap file allocation increment value.  The size in blocks to use to backup
; swap file space allocation in the swap or page file.  Space in the file will
; be allocated multiples of this unit up to wsquota to guarantee swap space.
;
        PARAMETER       ADDRESS=SWP$GW_SWPINC,-
                        DEFAULT=96,-
                        MIN=16,-
                        NAME=SWPALLOCINC,-
                        SIZE=WORD,-
                        TYPE=<SPECIAL>,-
                        UNIT=Blocks,-
                        VERSION_MASK=<SYSGEN>
;
;       I/O time allowance.  This parameter sets the number of 10 millisecond
;       units to charge the current residence quantum for each voluntary wait.
;       The correct value approximates the cost of a disk I/O neglecting wait time.
;
        PARAMETER       ADDRESS=SCH$GW_IOTA,-
                        DEFAULT=2,-
                        MIN=0,-
                        MAX=32767,-
                        NAME=IOTA,-
                        SIZE=WORD,-
                        TYPE=<SPECIAL,DYNAMIC>,-
                        UNIT=10Ms,-
                        VERSION_MASK=<SYSGEN>
;
;       Elapsed realtime to cause a HIB or LEF process to look like it is in
;       longwait.  This parameter sets the number of 1 second units
;       that need to have elapsed.  Longwait processes are one of the most
;       eligible to attempt to recover pages from when a shortage is detected.
;
        PARAMETER       ADDRESS=SCH$GW_LONGWAIT,-       ;
                        DEFAULT=30,-    ;
                        MIN=0,- ;
                        MAX=65535,-     ;
                        NAME=LONGWAIT,- ;
                        SIZE=WORD,-     ;
```

```
                        TYPE=<SYS,DYNAMIC>,-    ;
                        UNIT=Seconds,-
                        VERSION_MASK=<SYSGEN>

;       Elapsed realtime to cause a low priority COM process to look like it is
;       dormant. This parameter sets the number of 1 second units
;       that need to have elapsed.  Dormant processes are one of the most
;       eligible to attempt to recover pages from when a shortage is detected.
;       For V5.0+ systems, DORMANTWAIT is also used, in a compatible way, as a
;       control parameter for the new PIXSCAN processing algorithm.
;
        PARAMETER       ADDRESS=SCH$GW_DORMANTWAIT,-    ;
                        DEFAULT=2,-     ;
                        MIN=0,- ;
                        MAX=65535,-     ;
                        NAME=DORMANTWAIT,-      ;
                        SIZE=WORD,-     ;
                        TYPE=<SYS,DYNAMIC>,-    ;
                        UNIT=Seconds,-
                        VERSION_MASK=<SYSGEN>
;
;       Swap fail count. This parameter sets the number of consecutive swap
;       schedule failures to occur before the swap schedule algorithm changes
;       to ignore the swap quantum protection.
;
        PARAMETER       ADDRESS=SCH$GW_SWPFAIL,-        ;
                        DEFAULT=20,-    ;
                        MIN=0,- ;
                        MAX=32767,-     ;
                        NAME=SWPFAIL,-  ;
                        SIZE=WORD,-     ;
                        TYPE=<SPECIAL,DYNAMIC>,-
                        VERSION_MASK=<SYSGEN>

;
;       These are reserved parameters for undefined use by either Digital
;       or user written system services.
;
;
;       This is the start of the Digital reserved parameters.
;

        PARAMETER       ADDRESS=SGN$GL_VMSD1,-  ;
                        DEFAULT=0,-     ;
                        MIN=0,- ;
                        NAME=VMSD1,-    ;
                        SIZE=LONG,-     ;
                        TYPE=<SPECIAL,DYNAMIC>,-
                        VERSION_MASK=<SYSGEN>
;
        PARAMETER       ADDRESS=SGN$GL_VMSD2,-  ;
                        DEFAULT=0,-     ;
                        MIN=0,- ;
                        NAME=VMSD2,-    ;
                        SIZE=LONG,-     ;
                        TYPE=<SPECIAL,DYNAMIC>,-
                        VERSION_MASK=<SYSGEN>
```

```
2394 ;
2395          PARAMETER       ADDRESS=SGN$GL_VMSD3,-  ;
2396                          DEFAULT=0,-     ;
2397                          MIN=0,- ;
2398                          NAME=VMSD3,-    ;
2399                          SIZE=LONG,-     ;
2400                          TYPE=<SPECIAL,DYNAMIC>,-
2401                          VERSION_MASK=<SYSGEN>
2402 ;
2403          PARAMETER       ADDRESS=SGN$GL_VMSD4,-  ;
2404                          DEFAULT=0,-     ;
2405                          MIN=0,- ;
2406                          NAME=VMSD4,-    ;
2407                          SIZE=LONG,-     ;
2408                          TYPE=<SPECIAL,DYNAMIC>,-
2409                          VERSION_MASK=<SYSGEN>
2410 ;
2411          PARAMETER       ADDRESS=SGN$GL_VMS5,-   ;
2412                          DEFAULT=0,-     ;
2413                          MIN=0,- ;
2414                          NAME=VMS5,-     ;
2415                          SIZE=LONG,-     ;
2416                          TYPE=<SPECIAL>,-
2417                          VERSION_MASK=<SYSGEN>
2418 ;
2419          PARAMETER       ADDRESS=SGN$GL_VMS6,-   ;
2420                          DEFAULT=0,-     ;
2421                          MIN=0,- ;
2422                          NAME=VMS6,-     ;
2423                          SIZE=LONG,-     ;
2424                          TYPE=<SPECIAL>,-
2425                          VERSION_MASK=<SYSGEN>
2426 ;
2427          PARAMETER       ADDRESS=SGN$GL_VMS7,-   ;
2428                          DEFAULT=0,-     ;
2429                          MIN=0,- ;
2430                          NAME=VMS7,-     ;
2431                          SIZE=LONG,-     ;
2432                          TYPE=<SPECIAL>,-
2433                          VERSION_MASK=<SYSGEN>
2434 ;
2435          PARAMETER       ADDRESS=SGN$GL_VMS8,-   ;
2436                          DEFAULT=0,-     ;
2437                          MIN=0,- ;
2438                          NAME=VMS8,-     ;
2439                          SIZE=LONG,-     ;
2440                          TYPE=<SPECIAL>,-
2441                          VERSION_MASK=<SYSGEN>
2442
2443 ;
2444 ; Specify Job Controller options.
2445 ;
2446          PARAMETER       ADDRESS=SGN$GL_JOBCTLD,-        ;
2447                          DEFAULT=0,-     ;
2448                          MIN=0,- ;
2449                          NAME=JOBCTLD,-  ;
2450                          SIZE=LONG,-     ;
```

```
                        TYPE=<SPECIAL,DYNAMIC>,-
                        VERSION_MASK=<SYSGEN>

;
;  Specify PUDRIVER load options
;
        PARAMETER       ADDRESS=SGN$GL_PU_OPTIONS,-     ;
                        DEFAULT=0,-     ;
                        MIN=0,- ;
                        NAME=PU_OPTIONS,-       ;
                        SIZE=LONG,-     ;
                        TYPE=<SPECIAL,DYNAMIC>,-
                        VERSION_MASK=<SYSGEN>

;
;  Specify WPDRIVER options to control nonpaged pool allocation
;
        PARAMETER       ADDRESS=SGN$GL_WPTTE_SIZE,-     ;
                        DEFAULT=1000,-  ;
                        MIN=0,- ;
                        NAME=WPTTE_SIZE,-       ;
                        SIZE=LONG,-     ;
                        TYPE=<SPECIAL,DYNAMIC>,-;
                        UNIT=Entries,-
                        VERSION_MASK=<SYSGEN>,-
                        VERSION_MASK=<SYSGEN>

        PARAMETER       ADDRESS=SGN$GW_WPRE_SIZE,-      ;
                        DEFAULT=5,-     ;
                        MIN=0,- ;
                        NAME=WPRE_SIZE,-        ;
                        SIZE=WORD,-     ;
                        TYPE=<SPECIAL,DYNAMIC>,-;
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>

;
;       Qbus Multi-level interrupt option
;         enables software workaround in VECDEF (see: LOADER.MAR)
;
        PARAMETER       ADDRESS=SGN$GB_QBUS_MULT_INTR,- ;
                        DEFAULT=0,-     ;
                        MIN=0,-         ;
                        MAX=1,-         ;
                        NAME=QBUS_MULT_INTR,- ;
                        TYPE=<SPECIAL>,- ;
                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>


;
; Number of buffers in the Error Logging Buffer Ring
;
        PARAMETER       ADDRESS=SGN$GW_ERLBUFCNT,-
                        DEFAULT=4,-
                        MIN=2,-
                        MAX=64,-
```

```
2508                            NAME=ERRORLOGBUFFERS,-
2509                            SIZE=WORD,-
2510                            TYPE=<SYS>,-
2511                            UNIT=Buffers,-
2512                            VERSION_MASK=<SYSGEN>
2513
2514 ;
2515 ; Bitmask to specify the dump style options to follow in BUGCHECK.
2516 ;
2517 ;       The presently defined values are:
2518 ;
2519 ;               0       =       Full physical memory dump
2520 ;               1       =       Subset crash dump
2521 ;
2522                    PARAMETER       ADDRESS=SGN$GL_DUMP_STYLE,-
2523                                    DEFAULT=0,-
2524                                    MIN=0,-
2525                                    NAME=DUMPSTYLE,-
2526                                    SIZE=LONG,-
2527                                    TYPE=<SYS>,-
2528                                    UNIT=Bitmask,-
2529                                    VERSION_MASK=<SYSGEN>
2530
2531 ;
2532 ;       This is the start of the user reserved sysgen parameters.
2533 ;
2534            PARAMETER       ADDRESS=SGN$GL_USERD1,- ;
2535                            DEFAULT=0,-     ;
2536                            MIN=0,- ;
2537                            NAME=USERD1,-   ;
2538                            SIZE=LONG,-     ;
2539                            TYPE=<DYNAMIC>,-
2540                            VERSION_MASK=<SYSGEN>
2541 ;
2542            PARAMETER       ADDRESS=SGN$GL_USERD2,- ;
2543                            DEFAULT=0,-     ;
2544                            MIN=0,- ;
2545                            NAME=USERD2,-   ;
2546                            SIZE=LONG,-     ;
2547                            TYPE=<DYNAMIC>,-
2548                            VERSION_MASK=<SYSGEN>
2549 ;
2550            PARAMETER       ADDRESS=SGN$GL_USER3,-  ;
2551                            DEFAULT=0,-     ;
2552                            MIN=0,- ;
2553                            NAME=USER3,-    ;
2554                            SIZE=LONG,-     ;
2555                            TYPE=<>,-
2556                            VERSION_MASK=<SYSGEN>
2557 ;
2558            PARAMETER       ADDRESS=SGN$GL_USER4,-  ;
2559                            DEFAULT=0,-     ;
2560                            MIN=0,- ;
2561                            NAME=USER4,-    ;
2562                            SIZE=LONG,-     ;
2563                            TYPE=<>,-
2564                            VERSION_MASK=<SYSGEN>
```

```
2565
2566 ;
2567 ;       Extra CPU time.  This parameter sets the number of 10 millisecond
2568 ;       units to be allowed as an extension when CPU time expires.  One
2569 ;       extension is allowed for each access mode.
2570 ;
2571         PARAMETER       ADDRESS=SGN$GL_EXTRACPU,-       ;
2572                         DEFAULT=1000,-  ; 10 Seconds
2573                         MIN=0,-         ;
2574                         NAME=EXTRACPU,- ;
2575                         SIZE=LONG,-     ;
2576                         TYPE=<SYS,DYNAMIC>,-    ;
2577                         UNIT=10Ms,-
2578                         VERSION_MASK=<SYSGEN>
2579
2580 ;
2581 ; Maximum group code for system UIC
2582 ;
2583         PARAMETER       ADDRESS=EXE$GL_SYSUIC,- ;
2584                         DEFAULT=8,-     ;
2585                         MIN=1,-         ;
2586                         MAX=32768,-     ;
2587                         NAME=MAXSYSGROUP,-      ;
2588                         SIZE=LONG,-     ;
2589                         TYPE=<SYS,DYNAMIC>,-    ;
2590                         UNIT=<UIC Group>,-
2591                         VERSION_MASK=<SYSGEN>
2592 ;
2593 ; Maximum time for a device to languish in mount verification before giving up.
2594 ;
2595         PARAMETER       ADDRESS=IOC$GW_MVTIMEOUT,-
2596                         DEFAULT=3600,-                  ; 60 minute default
2597                         MIN=1,-
2598                         MAX=64000,-                     ;
2599                         NAME=MVTIMEOUT,-                ;
2600                         SIZE=WORD,-
2601                         TYPE=<SYS,DYNAMIC>,-
2602                         UNIT=Seconds,-
2603                         VERSION_MASK=<SYSGEN>
2604 ;
2605 ; Maximum time for a tape device to languish in mount verification before giving up.
2606 ;
2607         PARAMETER       ADDRESS=IOC$GW_TAPE_MVTIMEOUT,-
2608                         DEFAULT=600,-                   ; 10 minute default
2609                         MIN=1,-
2610                         MAX=64000,-                     ;
2611                         NAME=TAPE_MVTIMEOUT,-           ;
2612                         SIZE=WORD,-
2613                         TYPE=<SYS,DYNAMIC>,-
2614                         UNIT=Seconds,-
2615                         VERSION_MASK=<SYSGEN>
2616 ;
2617 ; Maximum allowable buffered I/O request size
2618 ;
2619         PARAMETER       ADDRESS=IOC$GW_MAXBUF,- ;
2620                         DEFAULT=2048,-  ;
2621                         MIN=1700,-      ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 50
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

2622                            MAX=64000,-     ;
2623                            NAME=MAXBUF,-   ;
2624                            SIZE=WORD,-     ;
2625                            TYPE=<SYS,DYNAMIC>,-    ;
2626                            UNIT=Bytes,-
2627                            VERSION_MASK=<SYSGEN>
2628 ;
2629 ; Default buffer quota for Mailbox creation
2630 ;
2631            PARAMETER        ADDRESS=IOC$GW_MBXBFQUO,-       ;
2632                            DEFAULT=1056,-  ;
2633                            MIN=256,-       ;
2634                            MAX=64000,-     ;
2635                            NAME=DEFMBXBUFQUO,-     ;
2636                            SIZE=WORD,-     ;
2637                            TYPE=<SYS,DYNAMIC>,-    ;
2638                            UNIT=Bytes,-
2639                            VERSION_MASK=<SYSGEN>
2640 ;
2641 ; Default maximum message size for Mailbox creation
2642 ;
2643            PARAMETER        ADDRESS=IOC$GW_MBXMXMSG,-       ;
2644                            DEFAULT=256,-   ;
2645                            MIN=64,-        ;
2646                            MAX=64000,-     ;
2647                            NAME=DEFMBXMXMSG,-      ;
2648                            SIZE=WORD,-     ;
2649                            TYPE=<SYS,DYNAMIC>,-    ;
2650                            UNIT=Bytes,-
2651                            VERSION_MASK=<SYSGEN>
2652 ;
2653 ;          DESIRED FREE LIST LENGTH - SPECIFIES THE NUMBER OF FREE PAGES TO
2654 ;                          BE MAINTAINED AVAILABLE BY THE SWAPPER.
2655 ;
2656            PARAMETER        ADDRESS=SGN$GL_FREELIM,-        ;
2657                            DEFAULT=32,-            ;
2658                            MIN=16,-                ;
2659                            NAME=FREELIM,-          ;
2660                            SIZE=LONG,-             ;
2661                            TYPE=<SYS,SYSGEN,MAJOR>,-       ;
2662                            UNIT=Pages,-
2663                            VERSION_MASK=<SYSGEN>
2664 ;
2665 ;          Target free list length - specifies the number of free pages
2666 ;                          that the swapper will attempt to make available
2667 ;                          when correcting for free list < FREELIM.
2668 ;
2669            PARAMETER        ADDRESS=SGN$GL_FREEGOAL,-       ;
2670                            DEFAULT=200,-           ;
2671                            MIN=16,-                ;
2672                            NAME=FREEGOAL,-         ;
2673                            SIZE=LONG,-             ;
2674                            TYPE=<SYS,MAJOR>,-      ;
2675                            UNIT=Pages,-
2676                            VERSION_MASK=<SYSGEN>
2677 ;
2678 ;          DESIRED FREE LIST LENGTH THAT MUST EXIST TO ALLOW PROCESSES
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 51
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

2679 ;                          TO GROW PAST WSQUOTA.
2680 ;
2681          PARAMETER         ADDRESS=SCH$GL_GROWLIM,-;
2682                            DEFAULT=63,-            ;
2683                            MIN=0,-                 ;
2684                            NAME=GROWLIM,-          ;
2685                            SIZE=LONG,-             ;
2686                            TYPE=<SYS,DYNAMIC,MAJOR>,-      ;
2687                            UNIT=Pages,-
2688                            VERSION_MASK=<SYSGEN>
2689 ;
2690 ;       DESIRED FREE LIST LENGTH THAT MUST EXIST TO ALLOW PROCESSES
2691 ;                          TO GROW PAST WSQUOTA.
2692 ;
2693          PARAMETER         ADDRESS=SCH$GL_BORROWLIM,-      ;
2694                            DEFAULT=300,-           ;
2695                            MIN=0,-                 ;
2696                            NAME=BORROWLIM,-        ;
2697                            SIZE=LONG,-             ;
2698                            TYPE=<SYS,DYNAMIC,MAJOR>,-      ;
2699                            UNIT=Pages,-
2700                            VERSION_MASK=<SYSGEN>
2701 ;
2702 ; NUMBER OF RETRIES TO PERFORM WHEN TRYING TO LOCK A MULTI-PROCESSOR
2703 ; DATA STRUCTURE
2704 ;
2705          PARAMETER         ADDRESS=EXE$GL_LOCKRTRY,-       ;
2706                            DEFAULT=100000,-        ;
2707                            MIN=1,-                 ;
2708                            NAME=LOCKRETRY,-        ;
2709                            SIZE=LONG,-             ;
2710                            TYPE=<SPECIAL,DYNAMIC>,-;
2711                            UNIT=Retries,-
2712                            VERSION_MASK=<SYSGEN>
2713 ;
2714 ; Maximum DR32 data rate
2715 ;
2716          PARAMETER         ADDRESS=IOC$GW_XFMXRATE,-       ;
2717                            DEFAULT=236,-           ;
2718                            MIN=0,-                 ;
2719                            MAX=255,-               ;
2720                            NAME=XFMAXRATE,-        ;
2721                            SIZE=WORD,-             ;
2722                            TYPE=<SYS,DYNAMIC>,-    ;
2723                            UNIT=Special,-
2724                            VERSION_MASK=<SYSGEN>
2725 ;
2726 ; Number of Unibus map registers to preallocate for LPA11
2727 ;
2728          PARAMETER         ADDRESS=IOC$GW_LAMAPREG,-       ;
2729                            DEFAULT=0,-             ;
2730                            MIN=0,-                 ;
2731                            MAX=255,-               ;
2732                            NAME=LAMAPREGS,-        ;
2733                            SIZE=WORD,-             ;
2734                            TYPE=<SYS,SYSGEN>,-             ;
2735                            UNIT=Mapregs,-
```

```
                        VERSION_MASK=<SYSGEN>

;
; Number of SPT entries to preallocate for use by real time processes
; connecting to devices via the connect to interrupt driver.
;

        PARAMETER       ADDRESS=EXE$GL_RTIMESPT,-
                        DEFAULT=0,-
                        MIN=0,-
                        NAME=REALTIME_SPTS,-
                        SIZE=LONG,-
                        TYPE=<SYS,SYSGEN>,-
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>

;
; Number of pages created for command interpreter symbol table.
;

        PARAMETER       ADDRESS=EXE$GL_CLITABL,-
                        DEFAULT=250,-
                        MIN=10,-
                        MAX=500,-
                        NAME=CLISYMTBL,-
                        SIZE=LONG,-
                        TYPE=<SYS,SYSGEN,DYNAMIC>,-
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>

;
; Initial size of lock id table (and growing increment).
;

        PARAMETER       ADDRESS=LCK$GL_IDTBLSIZ,-
                        DEFAULT=200,-
                        MIN=40,-
                        MAX=262143,-
                        NAME=LOCKIDTBL,-
                        SIZE=LONG,-
                        TYPE=<SYS,SYSGEN,MAJOR>,-
                        UNIT=Entries,-
                        VERSION_MASK=<SYSGEN>

;
; Maximum size of lock id table.
;

        PARAMETER       ADDRESS=LCK$GL_IDTBLMAX,-
                        DEFAULT=65535,-
                        MIN=200,-
                        MAX=262143,-
                        NAME=LOCKIDTBL_MAX,-
                        SIZE=LONG,-
                        TYPE=<SYS,SYSGEN,MAJOR,DYNAMIC>,-
                        UNIT=Entries,-
                        VERSION_MASK=<SYSGEN>
```

```
2793
2794 ;
2795 ; Size of resource hash table.
2796 ;
2797
2798         PARAMETER       ADDRESS=LCK$GL_HTBLSIZ,-
2799                         DEFAULT=64,-
2800                         MIN=1,-
2801                         MAX=8192,-
2802                         NAME=RESHASHTBL,-
2803                         SIZE=LONG,-
2804                         TYPE=<SYS,SYSGEN,MAJOR>,-
2805                         UNIT=Entries,-
2806                         VERSION_MASK=<SYSGEN>
2807
2808 ;
2809 ; Deadlock detection timeout period
2810 ;
2811
2812         PARAMETER       ADDRESS=LCK$GL_WAITTIME,-
2813                         DEFAULT=10,-
2814                         MIN=0,-
2815                         NAME=DEADLOCK_WAIT,-
2816                         SIZE=LONG,-
2817                         TYPE=<SYS,DYNAMIC>,-
2818                         UNIT=Seconds,-
2819                         VERSION_MASK=<SYSGEN>
2820
2821 ;
2822 ; SCS allocation counts - Buffer Descriptor Table entries
2823 ;
2824         PARAMETER       ADDRESS=SCS$GW_BDTCNT,-
2825                         DEFAULT=50,-
2826                         MIN=0,-
2827                         MAX=32767,-
2828                         NAME=SCSBUFFCNT,-
2829                         SIZE=WORD,-
2830                         TYPE=<SYSGEN,SCS>,-
2831                         UNIT=Entries,-
2832                         VERSION_MASK=<SYSGEN>
2833
2834 ;
2835 ; SCS allocation counts - Connect Descriptor Table entries
2836 ;
2837         PARAMETER       ADDRESS=SCS$GW_CDTCNT,-
2838                         DEFAULT=40,-
2839                         MIN=2,-
2840                         MAX=32767,-
2841                         NAME=SCSCONNCNT,-
2842                         SIZE=WORD,-
2843                         TYPE=<SYSGEN,SCS>,-
2844                         UNIT=Entries,-
2845                         VERSION_MASK=<SYSGEN>
2846
2847 ;
2848 ; SCS allocation counts - Response Descriptor Table entries
2849 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 54
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

2850          PARAMETER       ADDRESS=SCS$GW_RDTCNT,-
2851                          DEFAULT=300,-
2852                          MIN=0,-
2853                          MAX=32767,-
2854                          NAME=SCSRESPCNT,-
2855                          SIZE=WORD,-
2856                          TYPE=<SYSGEN,SCS>,-
2857                          UNIT=Entries,-
2858                          VERSION_MASK=<SYSGEN>
2859
2860 ;
2861 ; SCS maximum datagram size
2862 ;
2863
2864          PARAMETER       ADDRESS=SCS$GW_MAXDG,-
2865                          DEFAULT=576,-
2866                          MIN=28,-
2867                          MAX=985,-
2868                          NAME=SCSMAXDG,-
2869                          SIZE=WORD,-
2870                          TYPE=<SYSGEN,SCS>,-
2871                          UNIT=Bytes,-
2872                          VERSION_MASK=<SYSGEN>
2873
2874 ;
2875 ; SCS maximum sequenced message size
2876 ;
2877
2878          PARAMETER       ADDRESS=SCS$GW_MAXMSG,-
2879                          DEFAULT=132,-
2880                          MIN=52,-
2881                          MAX=985,-
2882                          NAME=SCSMAXMSG,-
2883                          SIZE=WORD,-
2884                          TYPE=<SYSGEN,SCS>,-
2885                          UNIT=Bytes,-
2886                          VERSION_MASK=<SYSGEN>
2887
2888 ;
2889 ; SCS flow control cushion
2890 ;
2891
2892          PARAMETER       ADDRESS=SCS$GW_FLOWCUSH,-
2893                          DEFAULT=1,-
2894                          MIN=0,-
2895                          MAX=16,-
2896                          NAME=SCSFLOWCUSH,-
2897                          SIZE=WORD,-
2898                          TYPE=<SCS,DYNAMIC>,-
2899                          UNIT=Credits,-
2900                          VERSION_MASK=<SYSGEN>
2901
2902 ;
2903 ; SCS system id (unique 48 bit number per system)
2904 ;
2905
2906          PARAMETER       ADDRESS=SCS$GB_SYSTEMID,-
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 55
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

                        DEFAULT=0,-
                        NAME=SCSSYSTEMID,-
                        SIZE=LONG,-
                        TYPE=<SYSGEN,SCS>,-
                        UNIT=Pure-number,-
                        VERSION_MASK=<SYSGEN>

        PARAMETER       ADDRESS=SCS$GB_SYSTEMIDH,-
                        DEFAULT=0,-
                        NAME=SCSSYSTEMIDH,-
                        SIZE=LONG,-
                        TYPE=<SYSGEN,SCS>,-
                        UNIT=Pure-number,-
                        VERSION_MASK=<SYSGEN>

;
; SCS system node name
;

        PARAMETER       ADDRESS=SCS$GB_NODENAME,-
                        DEFAULT=<^A/    />,-
                        MIN=<^A/    />,-
                        MAX=<^A/ZZZZ/>,-
                        NAME=SCSNODE,-
                        SIZE=QUAD,-
                        TYPE=<ASCII,SYSGEN,SCS>,-
                        UNIT=Ascii,-
                        VERSION_MASK=<SYSGEN>

;
; SCA process poller - polling interval
;

        PARAMETER       ADDRESS=SCS$GW_PRCPOLINT,-
                        DEFAULT=30,-
                        MIN=1,-
                        MAX=32767,-
                        NAME=PRCPOLINTERVAL,-
                        SIZE=WORD,-
                        TYPE=<SCS,DYNAMIC>,-
                        UNIT=Seconds,-
                        VERSION_MASK=<SYSGEN>

;
; CI port - timeout for START/STACK sequence, also basic driver wakeup interval
;

        PARAMETER       ADDRESS=SCS$GW_PASTMOUT,-
                        DEFAULT=5,-
                        MIN=1,-
                        MAX=99,-
                        NAME=PASTIMOUT,-
                        SIZE=WORD,-
                        TYPE=<SCS,DYNAMIC>,-
                        UNIT=Seconds,-
                        VERSION_MASK=<SYSGEN>
```

```
 2964 ;
 2965 ; CI port - number of DG buffers to queue for START handshake
 2966 ;
 2967
 2968         PARAMETER       ADDRESS=SCS$GW_PAPPDDG,-
 2969                         DEFAULT=4,-
 2970                         MIN=1,-
 2971                         MAX=16,-
 2972                         NAME=PASTDGBUF,-
 2973                         SIZE=WORD,-
 2974                         TYPE=<SCS>,-
 2975                         UNIT=Buffers,-
 2976                         VERSION_MASK=<SYSGEN>
 2977
 2978 ;
 2979 ; CI port - number of ports to poll each interval (for future expansion)
 2980 ;
 2981
 2982         PARAMETER       ADDRESS=SCS$GB_PANPOLL,-
 2983                         DEFAULT=16,-
 2984                         MIN=1,-
 2985                         MAX=223,-
 2986                         NAME=PANUMPOLL,-
 2987                         SIZE=BYTE,-
 2988                         TYPE=<SCS,DYNAMIC>,-
 2989                         UNIT=Ports,-
 2990                         VERSION_MASK=<SYSGEN>
 2991
 2992 ;
 2993 ; CI port - maximum port # to poll each interval (for future expansion)
 2994 ;
 2995
 2996         PARAMETER       ADDRESS=SCS$GB_PAMXPORT,-
 2997                         DEFAULT=64,-
 2998                         MIN=0,-
 2999                         MAX=223,-
 3000                         NAME=PAMAXPORT,-
 3001                         SIZE=BYTE,-
 3002                         TYPE=<SCS,DYNAMIC>,-
 3003                         UNIT=Port-number,-
 3004                         VERSION_MASK=<SYSGEN>
 3005 ;
 3006 ; CI port - time between poll initiates
 3007 ;
 3008
 3009         PARAMETER       ADDRESS=SCS$GW_PAPOLINT,-
 3010                         DEFAULT=5,-
 3011                         MIN=1,-
 3012                         MAX=32767,-
 3013                         NAME=PAPOLLINTERVAL,-
 3014                         SIZE=WORD,-
 3015                         TYPE=<SCS,DYNAMIC>,-
 3016                         UNIT=Seconds,-
 3017                         VERSION_MASK=<SYSGEN>
 3018
 3019 ;
 3020 ; CI port - time between check for SYSAP's waiting for pool
```

```
3021 ;
3022
3023         PARAMETER       ADDRESS=SCS$GW_PAPOOLIN,-
3024                         DEFAULT=15,-
3025                         MIN=1,-
3026                         MAX=32767,-
3027                         NAME=PAPOOLINTERVAL,-
3028                         SIZE=WORD,-
3029                         TYPE=<SCS,DYNAMIC>,-
3030                         UNIT=Seconds,-
3031                         VERSION_MASK=<SYSGEN>
3032
3033 ;
3034 ; CI port - Flags including sanity timer enable/disable
3035 ;
3036
3037         PARAMETER       ADDRESS=SCS$GB_PASANITY,-
3038                         DEFAULT=1,-
3039                         MIN=0,-
3040                         MAX=1,-
3041                         NAME=PASANITY,-
3042                         SIZE=BYTE,-
3043                         TYPE=<SCS,DYNAMIC>,-
3044                         UNIT=Boolean,-
3045                         VERSION_MASK=<SYSGEN>
3046
3047 ;
3048 ; CI port - Flags including CI remote port polling enable/disable
3049 ;
3050
3051         PARAMETER       ADDRESS=SCS$GB_PANOPOLL,-
3052                         DEFAULT=0,-
3053                         MIN=0,-
3054                         MAX=1,-
3055                         NAME=PANOPOLL,-
3056                         SIZE=BYTE,-
3057                         TYPE=<SCS,DYNAMIC>,-
3058                         UNIT=Boolean,-
3059                         VERSION_MASK=<SYSGEN>
3060
3061 ;
3062 ; This is the start of the PEDRIVER reserved SYSGEN parameters.
3063 ;
3064
3065         PARAMETER       ADDRESS=SGN$GL_PE1,-
3066                         DEFAULT=0,-
3067                         MIN=0,-
3068                         NAME=PE1,-
3069                         SIZE=LONG,-
3070                         TYPE=<SPECIAL,DYNAMIC>,-
3071                         VERSION_MASK=<SYSGEN>
3072
3073         PARAMETER       ADDRESS=SGN$GL_PE2,-
3074                         DEFAULT=0,-
3075                         MIN=0,-
3076                         NAME=PE2,-
3077                         SIZE=LONG,-
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 58
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

3078                            TYPE=<SPECIAL,DYNAMIC>,-
3079                            VERSION_MASK=<SYSGEN>
3080
3081            PARAMETER       ADDRESS=SGN$GL_PE3,-
3082                            DEFAULT=0,-
3083                            MIN=0,-
3084                            NAME=PE3,-
3085                            SIZE=LONG,-
3086                            TYPE=<SPECIAL,DYNAMIC>,-
3087                            VERSION_MASK=<SYSGEN>
3088
3089            PARAMETER       ADDRESS=SGN$GL_PE4,-
3090                            DEFAULT=0,-
3091                            MIN=0,-
3092                            NAME=PE4,-
3093                            SIZE=LONG,-
3094                            TYPE=<SPECIAL,DYNAMIC>,-
3095                            VERSION_MASK=<SYSGEN>
3096
3097            PARAMETER       ADDRESS=SGN$GL_PE5,-
3098                            DEFAULT=0,-
3099                            MIN=0,-
3100                            NAME=PE5,-
3101                            SIZE=LONG,-
3102                            TYPE=<SPECIAL>,-
3103                            VERSION_MASK=<SYSGEN>
3104
3105            PARAMETER       ADDRESS=SGN$GL_PE6,-
3106                            DEFAULT=0,-
3107                            MIN=0,-
3108                            NAME=PE6,-
3109                            SIZE=LONG,-
3110                            TYPE=<SPECIAL>,-
3111                            VERSION_MASK=<SYSGEN>
3112
3113 ;
3114 ;       Time prompt timeout - this parameter sets the amount of time to wait
3115 ;       for the time of day to be entered when booting.  The default value
3116 ;       of -1 gives the behavior of V2 and earlier.
3117 ;
3118            PARAMETER       ADDRESS=SGN$GW_TPWAIT,-
3119                            DEFAULT=-1,-
3120                            MIN=0,-
3121                            NAME=TIMEPROMPTWAIT,-
3122                            SIZE=WORD,-
3123                            TYPE=<SYS>,-
3124                            UNIT=uFortnights,-      ; Close enough to seconds
3125                            VERSION_MASK=<SYSGEN>
3126
3127 ;
3128 ; CLOCK_INTERVAL - Sets the number of microseconds between the hardware
3129 ;                  interval timer clock interrupts.
3130 ;
3131 ;                  This parameter has no effect on processors that have
3132 ;                  implemented only the subset interval clock registers.
3133 ;
3134            PARAMETER       ADDRESS=EXE$GW_CLKINT,-
```

```
                        DEFAULT=10000,-
                        MIN=500,-
                        MAX=10000,-
                        NAME=CLOCK_INTERVAL,-
                        SIZE=WORD,-
                        TYPE=<SYS,SPECIAL>,-
                        UNIT=us,-
                        VERSION_MASK=<SYSGEN>

;
; UDA port - UDABURSTRATE is one less than the maximum number of longwords
;            the host is willing to allow per NPR transfer.  Zero implies
;            the port should use its own default.  Both the port's
;            default and the maximum the port will accept are Controller
;            dependent.
;

        PARAMETER       ADDRESS=SCS$GB_UDABURST,-
                        DEFAULT=0,-
                        MIN=0,-
                        MAX=31,-
                        NAME=UDABURSTRATE,-
                        SIZE=BYTE,-
                        TYPE=<SYSGEN,SCS>,-
                        UNIT=Longwords,-
                        VERSION_MASK=<SYSGEN>


;
; NOTE: The following two entries must be contiguous and in order!!!!
;
; Size of SYSTEM space logical name hash table.
;

        PARAMETER       ADDRESS=LNM$GL_HTBLSIZS,-
                        DEFAULT=128,-
                        MIN=1,-
                        MAX=16383,-
                        NAME=LNMSHASHTBL,-
                        SIZE=LONG,-
                        TYPE=<SYS,SYSGEN>,-
                        UNIT=Entries,-
                        VERSION_MASK=<SYSGEN>

;
; Size of PROCESS space logical name hash table.
;

        PARAMETER       ADDRESS=LNM$GL_HTBLSIZP,-
                        DEFAULT=128,-
                        MIN=1,-
                        MAX=16383,-
                        NAME=LNMPHASHTBL,-
                        SIZE=LONG,-
                        TYPE=<SYS,SYSGEN>,-
                        UNIT=Entries,-
                        VERSION_MASK=<SYSGEN>
```

```
3192
3193          .IF     NOT_DEFINED VERSION
3194          .IF     NOT_DEFINED GETSYISW
3195 ;
3196 ;       PERMANENT DEFAULT SYSTEM FLAGS
3197 ;
3198          .ALIGN  LONG                    ;
3199          .IF     NDF,PRMSW               ;
3200 EXE$GL_DEFFLAGS::                       ;
3201          .ENDC                          ;
3202          .LONG   <1@EXE$V_POOLPGING>-    ; ENABLE SYSTEM POOL PAGING
3203                  !<1@EXE$V_SBIERR>-      ; SBI ERROR DETECTION
3204                  !<1@EXE$V_BUGREBOOT>-   ; AUTOMATIC REBOOT ON BUGCHECK
3205                  !<1@EXE$V_CRDENABL>-    ; ENABLE CRD ERROR DETECTION
3206                  !<1@EXE$V_BUGDUMP>-     ; SYSTEM DUMP ON BUGCHECK
3207                  !<1@EXE$V_FATAL_BUG>-   ; MAKE ALL BUGCHECKS FATAL
3208                  !<1@EXE$V_CONCEALED>-   ; ENABLE USE OF CONCEALED DEVICES
3209                  !<1@EXE$V_JOBQUEUES>-   ; Enable job controller queues
3210                  !<1@EXE$V_SHRF11ACP>    ; SHARE F11ACP
3211
3212          .ENDC   ; NOT_DEFINED GETSYISW
3213          .IF_FALSE
3214
3215          DEFINE  EXE$GL_DEFFLAGS,-
3216                  VERSION_MASK=<SYSGEN>
3217
3218          .ENDC   ; Version
3219
3220 ;
3221 ;       BUGCHECK REBOOT - ENABLES AUTOMATIC REBOOT ON BUGCHECK
3222 ;
3223          .NLIST  CND
3224          PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3225                          DEFAULT=1,-     ;
3226                          MAX=1,-         ;
3227                          MIN=0,-         ;
3228                          NAME=BUGREBOOT,-;
3229                          BIT=EXE$V_BUGREBOOT,-   ;
3230                          TYPE=<DYNAMIC,SYS>,-    ;
3231                          UNIT=Boolean,-
3232                          VERSION_MASK=<SYSGEN>
3233          .LIST   CND
3234 ;
3235 ;       CRD ERROR ENABLE - ENABLES DETECTION AND LOGGING OF MEMORY CRD ERRORS
3236 ;
3237          PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3238                          DEFAULT=1,-     ;
3239                          MAX=1,-         ;
3240                          MIN=0,-         ;
3241                          NAME=CRDENABLE,-;
3242                          BIT=EXE$V_CRDENABL,-;
3243                          TYPE=<SYS,SYSGEN>,-;
3244                          UNIT=Boolean,-
3245                          VERSION_MASK=<SYSGEN>
3246 ;
3247 ;       BUGCHECK DUMP ENABLE - ENABLE SYSTEM DUMP ON BUGCHECK
3248 ;
```

```
3249          PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3250                          DEFAULT=1,-     ;
3251                          MAX=1,-         ;
3252                          MIN=0,-         ;
3253                          NAME=DUMPBUG,-  ;
3254                          BIT=EXE$V_BUGDUMP,-     ;
3255                          TYPE=<SYS>,-
3256                          UNIT=Boolean,-
3257                          VERSION_MASK=<SYSGEN>
3258 ;
3259 ;       FATAL BUGCHECK - TURNS ALL CONTINUABLE BUGCHECKS INTO FATAL BUGCHECKS
3260 ;
3261          PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3262                          DEFAULT=0,-     ;
3263                          MAX=1,-         ;
3264                          MIN=0,-         ;
3265                          NAME=BUGCHECKFATAL,-;
3266                          BIT=EXE$V_FATAL_BUG,-   ;
3267                          TYPE=<SYS,DYNAMIC>,-;
3268                          UNIT=Boolean,-
3269                          VERSION_MASK=<SYSGEN>
3270 ;
3271 ;       MULTIPLE ACP - SPECIFIES THAT SEPARATE ACPS ARE TO BE CREATED FOR
3272 ;                      EACH CLASS OF DISK.
3273
3274          PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-;
3275                          DEFAULT=0,-     ;
3276                          MAX=1,-         ;
3277                          MIN=0,-         ;
3278                          NAME=ACP_MULTIPLE,-     ;
3279                          BIT=EXE$V_MULTACP,-     ;
3280                          TYPE=<ACP,DYNAMIC>,-    ;
3281                          UNIT=Boolean,-
3282                          VERSION_MASK=<SYSGEN>
3283 ;
3284 ;       AUTO CONFIGURATION INHIBIT - INHIBITS THE AUTOMATIC CONFIGURATION
3285 ;               OF DEVICES.
3286 ;
3287          PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-;
3288                          DEFAULT=0,-     ;
3289                          MAX=1,-         ;
3290                          MIN=0,-         ;
3291                          NAME=NOAUTOCONFIG,-
3292                          BIT=EXE$V_NOAUTOCNF,-   ;
3293                          TYPE=<SPECIAL,DYNAMIC>,-;
3294                          UNIT=Boolean,-
3295                          VERSION_MASK=<SYSGEN>
3296 ;
3297 ;       NO CLOCK - INHIBITS VARIOUS TIME-RELATED FUNCTIONS
3298 ;
3299          PARAMETER       ADDRESS=EXE$GL_TIME_CONTROL,-   ;
3300                          DEFAULT=0,-     ;
3301                          MAX=-1,-        ;
3302                          MIN=0,-         ;
3303                          NAME=TIME_CONTROL,-     ;
3304                          TYPE=<SPECIAL,DYNAMIC>,-        ;
3305                          UNIT=Bit-mask,-
```

```
3306                            VERSION_MASK=<SYSGEN>
3307 ;
3308 ;       BREAKPOINTS - BITMASK WHICH ALLOWS WHICH HARDCODED
3309 ;               CALLERS OF THE INITIAL BREAKPOINT IN THE
3310 ;               EXECUTIVE ARE ALLOWED TO MAKE THAT CALL.
3311 ;
3312            PARAMETER           ADDRESS=SGN$GL_BRKMSK,-         ;
3313                                DEFAULT=<BPT$M_INITBEGIN!-      ;
3314                                        BPT$M_INITEND>,-        ;
3315                                MAX=-1,-                        ;
3316                                MIN=0,-                         ;
3317                                NAME=BREAKPOINTS,-              ;
3318                                TYPE=<SPECIAL>,-                ;
3319                                UNIT=Bitmask,-
3320                                VERSION_MASK=<SYSGEN>
3321 ;
3322 ;       CLUSTERING INHIBIT - INHIBITS ALL PAGE READ CLUSTERING
3323 ;
3324            PARAMETER           ADDRESS=EXE$GL_DEFFLAGS,-       ;
3325                                DEFAULT=0,-     ;
3326                                MAX=1,-         ;
3327                                MIN=0,-         ;
3328                                NAME=NOCLUSTER,-        ;
3329                                BIT=EXE$V_NOCLUSTER,-   ;
3330                                TYPE=<SPECIAL>,-        ;
3331                                UNIT=Boolean,-
3332                                VERSION_MASK=<SYSGEN>
3333 ;
3334 ;       ENABLE PAGING OF PAGED DYNAMIC POOL
3335 ;
3336            PARAMETER           ADDRESS=EXE$GL_DEFFLAGS,-       ;
3337                                DEFAULT=1,-     ;
3338                                MAX=1,-         ;
3339                                MIN=0,-         ;
3340                                NAME=POOLPAGING,-       ;
3341                                BIT=EXE$V_POOLPGING,-   ;
3342                                TYPE=<SPECIAL>,-;
3343                                UNIT=Boolean,-
3344                                VERSION_MASK=<SYSGEN>
3345 ;
3346 ;       SBI ERROR DETECTION ENABLE
3347 ;
3348            PARAMETER           ADDRESS=EXE$GL_DEFFLAGS,-       ;
3349                                DEFAULT=1,-     ;
3350                                MAX=1,-         ;
3351                                MIN=0,-         ;
3352                                NAME=SBIERRENABLE,-     ;
3353                                BIT=EXE$V_SBIERR,-      ;
3354                                TYPE=<SPECIAL>,-        ;
3355                                UNIT=Boolean,-
3356                                VERSION_MASK=<SYSGEN>
3357 ;
3358 ;       FORCE ENTRY OF TIME AT SYSTEM BOOT
3359 ;
3360            PARAMETER           ADDRESS=EXE$GL_DEFFLAGS,-       ;
3361                                DEFAULT=0,-     ;
3362                                MAX=1,-         ;
```

```
3363                            MIN=0,-         ;
3364                            NAME=SETTIME,-  ;
3365                            BIT=EXE$V_SETTIME,-     ;
3366                            TYPE=<SYS,SYSGEN>,-;
3367                            UNIT=Boolean,-
3368                            VERSION_MASK=<SYSGEN>
3369 ;
3370 ;       ENABLE SHARING OF F11ACP
3371 ;
3372            PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3373                            DEFAULT=1,-     ;
3374                            MAX=1,-         ;
3375                            MIN=0,-         ;
3376                            NAME=ACP_SHARE,-;
3377                            BIT=EXE$V_SHRF11ACP,-   ;
3378                            TYPE=<ACP>,-;
3379                            UNIT=Boolean,-
3380                            VERSION_MASK=<SYSGEN>
3381 ;
3382 ;       SELECT ALTERNATE AUTHORIZATION FILE - CAUSES SYSINIT TO MAKE A LOGICAL
3383 ;               NAME REDIRECTING SYSUAF TO SYSUAFALT.
3384 ;
3385            PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3386                            DEFAULT=0,-     ;
3387                            MAX=1,-         ;
3388                            MIN=0,-         ;
3389                            NAME=UAFALTERNATE,-     ;
3390                            BIT=EXE$V_SYSUAFALT,-   ;
3391                            TYPE=<SYS,SYSGEN>,-     ;
3392                            UNIT=Boolean,-
3393                            VERSION_MASK=<SYSGEN>
3394 ;
3395 ;       LEAVE SYSTEM WRITABLE - FOR DEBUGGING PURPOSES LEAVES SYSTEM CODE
3396 ;               WRITABLE.
3397            PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3398                            DEFAULT=0,-     ;
3399                            MAX=1,-         ;
3400                            MIN=0,-         ;
3401                            NAME=WRITABLESYS,-;
3402                            BIT=EXE$V_SYSWRTABL,-;
3403                            TYPE=<SPECIAL>,-        ;
3404                            UNIT=Boolean,-
3405                            VERSION_MASK=<SYSGEN>
3406 ;
3407 ;
3408 ;       Enable resource allocation checking
3409 ;
3410
3411            PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3412                            DEFAULT=0,-                     ;
3413                            MAX=1,-                         ;
3414                            MIN=0,-                         ;
3415                            NAME=RESALLOC,-                 ;
3416                            BIT=EXE$V_RESALLOC,-            ;
3417                            TYPE=<SPECIAL>,-                ;
3418                            UNIT=Boolean,-
3419                            VERSION_MASK=<SYSGEN>
```

```
3420
3421 ;
3422 ;       SET TO INHIBIT SYSTEM SERVICES ON A PER PROCESS BASIS
3423 ;
3424         PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3425                         DEFAULT=0,-     ;
3426                         MAX=1,-         ;
3427                         MIN=0,-         ;
3428                         NAME=SSINHIBIT,-;
3429                         BIT=EXE$V_SSINHIBIT,-;
3430                         TYPE=<SPECIAL>,-        ;
3431                         UNIT=Boolean,-
3432                         VERSION_MASK=<SYSGEN>
3433
3434 ;
3435 ;       RESET TO DISABLE THE USE OF CONCEALED DEVICES
3436 ;
3437         PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3438                         DEFAULT=1,-     ;
3439                         MAX=1,-         ;
3440                         MIN=0,-         ;
3441                         NAME=CONCEAL_DEVICES,-;
3442                         BIT=EXE$V_CONCEALED,-;
3443                         TYPE=<SPECIAL>,-        ;
3444                         UNIT=Boolean,-
3445                         VERSION_MASK=<SYSGEN>
3446 ;
3447 ;       SAVEDUMP - IF THE DUMP IS IN THE PAGE FILE, SAVE IT UNTIL IT IS
3448 ;               ANALYZED AND COPIED.
3449 ;
3450         PARAMETER       ADDRESS=EXE$GL_DEFFLAGS,-       ;
3451                         DEFAULT=0,-     ;
3452                         MAX=1,-         ;
3453                         MIN=0,-         ;
3454                         NAME=SAVEDUMP,- ;
3455                         BIT=EXE$V_SAVEDUMP,-    ;
3456                         TYPE=<SYS>,-    ;
3457                         UNIT=Boolean,-
3458                         VERSION_MASK=<SYSGEN>
3459
3460         .IF     NOT_DEFINED VERSION
3461         .IF     NOT_DEFINED GETSYISW
3462 ;
3463 ; DYNAMIC SYSTEM CONTROL FLAGS.
3464 ;
3465         .ALIGN  LONG                    ;
3466         .IF     NDF,PRMSW               ;
3467 EXE$GL_DYNAMIC_FLAGS::                  ; Dynamic SYSGEN flags
3468         .ENDC   ; NDF,PRMSW
3469         .LONG   <1@EXE$V_WRITESYSPARAMS>!-
3470                 <1@EXE$V_BRK_TERM>;
3471         .ENDC   ; NOT_DEFINED GETSYISW
3472
3473         $VIELD  EXE,0,<-                ; DEFINITION FOR EXE$GL_DYNAMIC_FLAGS
3474                 CLASS_PROT,-            ; Do non-discretionary classification check
3475                 WRITESYSPARAMS,-        ; Write the active parameters to the system
3476                 BRK_TERM,-              ; Associate on terminal in breakin detection
```

```
                BRK_DISUSER,-           ; Disable user account on breakin
                NOPGFLSWP,-             ; Disallow swapping into page files
        >

        .IF_FALSE

        DEFINE  EXE$GL_DYNAMIC_FLAGS,-
                VERSION_MASK=<SYSGEN>

        DEFINE  EXE$V_CLASS_PROT,-
                VERSION_MASK=<SYSGEN>

        DEFINE  EXE$V_WRITESYSPARAMS,-
                VERSION_MASK=<SYSGEN>

        DEFINE  EXE$V_BRK_TERM,-
                VERSION_MASK=<SYSGEN>

        DEFINE  EXE$V_BRK_DISUSER,-
                VERSION_MASK=<SYSGEN>

        DEFINE  EXE$V_NOPGFLSWP,-
                VERSION_MASK=<SYSGEN>


        .ENDC

;
;       CLASS_PROT -    Perform the non-discretionary classification
;                       checks.  This also is looked at by the XQP to
;                       determine of a classification block should be
;                       added to the header of any created files.
;
        PARAMETER       ADDRESS=EXE$GL_DYNAMIC_FLAGS,-  ;
                        DEFAULT=0,-     ;
                        MAX=1,-         ;
                        MIN=0,-         ;
                        NAME=CLASS_PROT,-;
                        BIT=EXE$V_CLASS_PROT,-  ;
                        TYPE=<DYNAMIC,SYS>,-    ;
                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>

;
;
;       WRITESYSPARAMS - Set by SYSBOOT if a USE DEFAULT, USE FILE, or a
;                        SET command is executed. Cleared if a USE CURRENT or
;                        WRITE CURRENT command is executed. If set STARTUP.COM
;                        will issue a WRITE CURRENT SYSGEN command.
;
        PARAMETER       ADDRESS=EXE$GL_DYNAMIC_FLAGS,-  ;
                        DEFAULT=1,-     ;
                        MAX=1,-         ;
                        MIN=0,-         ;
                        NAME=WRITESYSPARAMS,-;
                        BIT=EXE$V_WRITESYSPARAMS,-;
                        TYPE=<DYNAMIC,SPECIAL>,-;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 66
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>

;
;       LGI_BRK_TERM -  Use the terminal name in the association string
;                       used in LOGIN's breakin detection.  If not set,
;                       breakin detection associates on username alone
;                       for terminal logins.
;
        PARAMETER       ADDRESS=EXE$GL_DYNAMIC_FLAGS,-  ;
                        DEFAULT=1,-     ;
                        MAX=1,-         ;
                        MIN=0,-         ;
                        NAME=LGI_BRK_TERM,-;
                        BIT=EXE$V_BRK_TERM,-;
                        TYPE=<DYNAMIC,LGI>,-;
                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>

;
;       LGI_BRK_DISUSER - If enabled, set the DISUSER flag in the user's
;                       UAF record if a breakin attempt is detected.
;                       This assures a permanent lockout of the user
;                       until re-enabled by the system manager.
;
        PARAMETER       ADDRESS=EXE$GL_DYNAMIC_FLAGS,-  ;
                        DEFAULT=0,-     ;
                        MAX=1,-         ;
                        MIN=0,-         ;
                        NAME=LGI_BRK_DISUSER,-;
                        BIT=EXE$V_BRK_DISUSER,-;
                        TYPE=<DYNAMIC,LGI>,-;
                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>


;
;       NOPGFLSWP - If set, disable swapping into pagefiles
;
        PARAMETER       ADDRESS=EXE$GL_DYNAMIC_FLAGS,-  ;
                        DEFAULT=0,-     ;
                        MAX=1,-         ;
                        MIN=0,-         ;
                        NAME=NOPGFLSWP,-;
                        BIT=EXE$V_NOPGFLSWP,-;
                        TYPE=<DYNAMIC,SPECIAL>,-;
                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>

        .IF     NOT_DEFINED VERSION
        .IF     NOT_DEFINED GETSYISW
;
; STATIC SYSTEM CONTROL FLAGS.
;
        .ALIGN  LONG                    ;
        .IF     NDF,PRMSW               ;
EXE$GL_STATIC_FLAGS::                   ; Static SYSGEN flags
```

```
    SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 67
    X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

     3591          .ENDC    ; NDF,PRMSW
     3592          .LONG    <1@EXE$V_XQP_RESIDENT>!<1@EXE$V_REBLDSYSD>
     3593          .ENDC    ; NOT_DEFINED GETSYISW
     3594
     3595          $VIELD   EXE,0,<-                ; DEFINITION FOR EXE$GL_STATIC_FLAGS
     3596                   XQP_RESIDENT,-          ; MEMORY RESIDENT XQP
     3597                   REBLDSYSD,-             ; REBUILD SYSTEM DISK IN SYSMOUNT
     3598                   SHADOWING,-             ; LOAD VOLUME SHADOWING CODE
     3599                   SA_APP,-                ; Booting stand-alone application (SA-
BACKUP
     3600          >
     3601
     3602          .IF_FALSE
     3603
     3604          DEFINE   EXE$GL_STATIC_FLAGS,-
     3605                   VERSION_MASK=<SYSGEN>
     3606
     3607          DEFINE   EXE$V_XQP_RESIDENT,-
     3608                   VERSION_MASK=<SYSGEN>
     3609
     3610          DEFINE   EXE$V_REBLDSYSD,-
     3611                   VERSION_MASK=<SYSGEN>
     3612
     3613          DEFINE   EXE$V_SHADOWING,-
     3614                   VERSION_MASK=<SYSGEN>
     3615
     3616          .ENDC    ; Version
     3617
     3618 ;
     3619 ;        SHADOWING - If set, use shadowing Disk Class Driver (DSDRIVER)
     3620 ;                    If clear, use regular Disk Class Driver (DUDRIVER)
     3621 ;
     3622          PARAMETER        ADDRESS=EXE$GL_STATIC_FLAGS,-
     3623                           DEFAULT=0,-
     3624                           MIN=0,-
     3625                           MAX=1,-
     3626                           NAME=SHADOWING,-
     3627                           BIT=EXE$V_SHADOWING,-
     3628                           TYPE=<SYS,STATIC>,-
     3629                           UNIT=Boolean,-
     3630                           VERSION_MASK=<SYSGEN>
     3631
     3632 ;
     3633          .PAGE
     3634          .SBTTL   SYSTEM MESSAGE PARAMETERS
     3635 ;
     3636 ; DEFINE THE CONTROL BITS IN EXE$GL_MSGFLAGS
     3637 ;
     3638          .IF      NOT_DEFINED VERSION
     3639
     3640          $GBLINI  GLOBAL
     3641          $VIELD   EXE,0,<-
     3642                   MOUNTMSG,-              ; ENABLE MOUNT NOTIFICATION
     3643                   DISMOUMSG,-             ; ENABLE DISMOUNT NOTIFICATION
     3644          >
     3645
     3646          .IF      NOT_DEFINED GETSYISW
     3647 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 68
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 3648 ; DEFINE THE EXE$GL_MSGFLAGS LONGWORD AND ITS INITIAL VALUE.
 3649 ;
 3650         .ALIGN  LONG                    ;
 3651         .IF     NDF,PRMSW               ;
 3652 EXE$GL_MSGFLAGS::                       ;
 3653         .ENDC                           ;
 3654         .LONG   0
 3655
 3656         .ENDC   ; NOT_DEFINED GETSYISW
 3657         .IF_FALSE
 3658
 3659         DEFINE  EXE$GL_MSGFLAGS,-
 3660                 VERSION_MASK=<SYSGEN>
 3661
 3662         DEFINE  EXE$V_MOUNTMSG,-
 3663                 VERSION_MASK=<SYSGEN>
 3664
 3665         DEFINE  EXE$V_DISMOUMSG,-
 3666                 VERSION_MASK=<SYSGEN>
 3667
 3668         .ENDC   ; Version
 3669
 3670 ;
 3671 ;       MOUNTMSG - CONTROLS OPERATOR NOTIFICATION OF VOLUME MOUNTING
 3672 ;
 3673         PARAMETER       ADDRESS=EXE$GL_MSGFLAGS,-       ;
 3674                         DEFAULT=0,-     ;
 3675                         MAX=1,-         ;
 3676                         MIN=0,-         ;
 3677                         NAME=MOUNTMSG,- ;
 3678                         BIT=EXE$V_MOUNTMSG,-    ;
 3679                         TYPE=<SYS,DYNAMIC>,-    ;
 3680                         UNIT=Boolean,-
 3681                         VERSION_MASK=<SYSGEN>
 3682 ;
 3683 ;       DISMOUMSG - Controls operator notification of volume dismounting
 3684 ;
 3685         PARAMETER       ADDRESS=EXE$GL_MSGFLAGS,-       ;
 3686                         DEFAULT=0,-     ;
 3687                         MAX=1,-         ;
 3688                         MIN=0,-         ;
 3689                         NAME=DISMOUMSG,-        ;
 3690                         BIT=EXE$V_DISMOUMSG,-   ;
 3691                         TYPE=<SYS,DYNAMIC>,-    ;
 3692                         UNIT=Boolean,-
 3693                         VERSION_MASK=<SYSGEN>
 3694         .PAGE
 3695         .SBTTL  SYSTEM LOADABLE CODE PARAMETERS
 3696 ;
 3697 ; DEFINE THE CONTROL BITS IN SGN$GL_LOADFLAGS
 3698 ;
 3699         .IF     NOT_DEFINED VERSION
 3700         $GBLINI GLOBAL
 3701         $VIELD  SGN,0,<-
 3702                 LOAD_SYS_IMAGES,-       ; LOAD SYSTEM IMAGES
 3703         >
 3704
```

```
3705          .IF     NOT_DEFINED GETSYISW
3706
3707 ;
3708 ; DEFINE THE SGN$GL_LOADFLAGS LONGWORD AND ITS INITIAL VALUE.
3709 ;
3710          .ALIGN  LONG                    ;
3711          .IF     NDF,PRMSW               ;
3712 SGN$GL_LOADFLAGS::                       ;
3713          .ENDC                           ;
3714
3715          .LONG   <1@SGN$V_LOAD_SYS_IMAGES> ; ENABLE LOADING OF SYSTEM IMAGES
3716
3717          .ENDC   ; NOT_DEFINED GETSYISW
3718          .IF_FALSE
3719
3720          DEFINE  SGN$GL_LOADFLAGS,-
3721                  VERSION_MASK=<SYSGEN>
3722
3723          DEFINE  SGN$V_LOAD_SYS_IMAGES,-
3724                  VERSION_MASK=<SYSGEN>
3725
3726          .ENDC   ; Version
3727
3728
3729 ;
3730 ;       LOAD_SYS_IMAGES - CONTROLS LOADING SYSTEM IMAGES
3731 ;
3732          PARAMETER       ADDRESS=SGN$GL_LOADFLAGS,-
3733                          DEFAULT=1,-
3734                          MAX=1,-
3735                          MIN=0,-
3736                          NAME=LOAD_SYS_IMAGES,-
3737                          BIT=SGN$V_LOAD_SYS_IMAGES,-
3738                          TYPE=<SPECIAL>,-
3739                          UNIT=Boolean,-
3740                          VERSION_MASK=<SYSGEN>
3741          .PAGE
3742          .SBTTL  TERMINAL DRIVER SYSTEM PARAMETERS
3743 ;
3744 ;       DIALUP SUPPORT CONTROL PARAMETERS
3745 ;
3746 ; DELTA TIME FOR DIALUP TIMER SCAN
3747 ;
3748          PARAMETER       ADDRESS=TTY$GL_DELTA,-  ;
3749                          DEFAULT=<100000*100>,-  ;
3750                          MIN=100000,-            ;
3751                          NAME=TTY_SCANDELTA,-    ;
3752                          SIZE=LONG,-             ;
3753                          TYPE=<TTY>,-            ;
3754                          UNIT=100Ns,-
3755                          VERSION_MASK=<SYSGEN>
3756 ;
3757 ; FLAGS FOR DIALUP
3758 ;
3759 ;       BIT 0 is 0 => NORMAL, 1 => UNITED KINGDOM
3760 ;       BIT 1 SPECIFIES ALTERNATE MODEM PROTOCOL
3761 ;
```

```
3762          PARAMETER       ADDRESS=TTY$GB_DIALTYP,-        ;
3763                          DEFAULT=0,-             ;
3764                          MIN=0,-                 ;
3765                          MAX=<^X0FF>,-           ;
3766                          NAME=TTY_DIALTYPE,-     ;
3767                          TYPE=<TTY>,-            ;
3768                          SIZE=BYTE,-             ;
3769                          UNIT=Bit-Encoded,-
3770                          VERSION_MASK=<SYSGEN>
3771 ;
3772 ; NOTE ALIGNMENT!
3773 ;
3774 ; TERMINAL CANONICAL CHARACTERISTICS
3775 ;
3776 ;
3777 ;       DEFAULT SPEED FOR TERMINALS
3778 ;
3779          PARAMETER       ADDRESS=TTY$GB_DEFSPEED,-; DEFAULT SPEED FOR TERMINALS AND P
3780                          DEFAULT=TT$C_BAUD_9600,-; 9600 BAUD - NO PARITY
3781                          MIN=1,-                 ;
3782                          MAX=16,-                ;
3783                          NAME=TTY_SPEED,-        ;
3784                          SIZE=BYTE,-             ;
3785                          TYPE=<TTY>,-            ;
3786                          UNIT=Special,-          ; TT$C_BAUD VALUES
3787                          VERSION_MASK=<SYSGEN>
3788 ;
3789 ;       DEFAULT RECEIVE SPEED
3790 ;
3791          PARAMETER       ADDRESS=TTY$GB_RSPEED,- ;THE RECEIVE SPEED FOR A TERMINAL
3792                          DEFAULT=0,-             ;USE THE DEFAULT SPEED
3793                          MIN=0,-                 ;
3794                          MAX=16,-                ;
3795                          NAME=TTY_RSPEED,-       ;
3796                          SIZE=BYTE,-             ;
3797                          TYPE=<TTY>,-            ;
3798                          UNIT=Special,-          ; TT$C_BAUD VALUES AND 0
3799                          VERSION_MASK=<SYSGEN>
3800 ;
3801 ;       DEFAULT PARITY
3802 ;
3803          PARAMETER       ADDRESS=TTY$GB_PARITY,- ;THE PARITY OF THE TERMINALS
3804                          DEFAULT=24,-            ;NO PARITY EIGHT BITS.
3805                          MIN=0,-                 ;
3806                          MAX=<^X0FF>,-           ;
3807                          NAME=TTY_PARITY,-       ;
3808                          SIZE=BYTE,-             ;
3809                          TYPE=<TTY>,-            ;
3810                          UNIT=Special,-          ;
3811                          VERSION_MASK=<SYSGEN>
3812 ;
3813 ;       DEFAULT TERMINAL LINE WIDTH
3814 ;
3815          PARAMETER       ADDRESS=TTY$GW_DEFBUF,- ; DEFAULT BUFFER SIZE
3816                          DEFAULT=80,-    ;
3817                          MIN=0,-         ;
3818                          MAX=65535,-     ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 71
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

                        NAME=TTY_BUF,-  ;
                        SIZE=WORD,-     ;
                        TYPE=<TTY>,-    ;
                        UNIT=Characters,-
                        VERSION_MASK=<SYSGEN>
;
;       DEFAULT TERMINAL CHARACTERISTICS
;
        PARAMETER       ADDRESS=TTY$GL_DEFCHAR,-; DEFAULT CHARACTERISTICS
        DEFAULT=<<24@TT$V_PAGE>+TT$M_TTSYNC+TT$M_WRAP+TT$M_LOWER+TT$M_SCOPE>,-

                        MIN=0,-                 ;
                        NAME=TTY_DEFCHAR,-      ;
                        SIZE=LONG,-             ;
                        TYPE=<TTY>,-            ;
                        UNIT=Bit-Encoded,-
                        VERSION_MASK=<SYSGEN>
;
; Device characteristics second word.
;
        PARAMETER       ADDRESS=TTY$GL_DEFCHAR2,-;
                        DEFAULT=TT2$M_EDITING!TT2$M_AUTOBAUD,-; SET AUTOBAUD FOR DEF
                        MIN=0,-
                        NAME=TTY_DEFCHAR2,-
                        SIZE=LONG,-
                        TYPE=<TTY>,-
                        UNIT=Bit-Encoded,-
                        VERSION_MASK=<SYSGEN>

;
; SIZE OF TYPEAHEAD BUFFER
;
        PARAMETER       ADDRESS=TTY$GW_TYPAHDSZ,-       ;
                        DEFAULT=78,-                    ;
                        MIN=0,-                         ;
                        NAME=TTY_TYPAHDSZ,-             ;
                        SIZE=WORD,-                     ;
                        TYPE=<TTY>,-                    ;
                        UNIT=Bytes,-
                        VERSION_MASK=<SYSGEN>
;
; Alternate Typeahead size.
;
        PARAMETER       ADDRESS=TTY$GW_ALTYPAHD,-       ;
                        DEFAULT=200,-                   ;
                        MIN=0,-
                        MAX=<^X07FFF>,-         ;
                        NAME=TTY_ALTYPAHD,-
                        SIZE=WORD,-
                        TYPE=<TTY>,-
                        UNIT=Bytes,-
                        VERSION_MASK=<SYSGEN>
;
; Alternate Typehaead buffer alarm size.
;
        PARAMETER       ADDRESS=TTY$GW_ALTALARM,-       ;
                        DEFAULT=64,-                    ;ASSUME WORST CASE
                        MIN=0,-
```

```
3876                            NAME=TTY_ALTALARM,-
3877                            SIZE=WORD,-
3878                            TYPE=<TTY>,-
3879                            UNIT=Bytes,-
3880                            VERSION_MASK=<SYSGEN>
3881 ;
3882 ; DMA size
3883 ;
3884            PARAMETER        ADDRESS=TTY$GW_DMASIZE,-        ;
3885                            DEFAULT=64,-
3886                            MIN=0,-
3887                            NAME=TTY_DMASIZE,-
3888                            SIZE=WORD,-
3889                            TYPE=<TTY,DYNAMIC>,-
3890                            UNIT=Bytes,-
3891                            VERSION_MASK=<SYSGEN>
3892
3893 ;
3894 ; DEFAULT TERMINAL ALLOCATION PROTECTION
3895 ;
3896            PARAMETER        ADDRESS=TTY$GW_PROT,-           ; PROTECTION CLASSES
3897                            DEFAULT=<^X0FFF0>,-             ; SYSTEM ONLY
3898                            MIN=0,-                         ;
3899                            NAME=TTY_PROT,-                 ;
3900                            SIZE=WORD,-                     ;
3901                            TYPE=<TTY>,-                    ;
3902                            UNIT=Protection,-                       ;
3903                            VERSION_MASK=<SYSGEN>
3904
3905            PARAMETER        ADDRESS=TTY$GL_OWNUIC,-         ; OWNER UIC
3906                            DEFAULT=<^X00010004>,-          ; SYSTEM OWNER
3907                            MIN=0,-                         ;
3908                            NAME=TTY_OWNER,-                ;
3909                            SIZE=LONG,-                     ;
3910                            TYPE=<TTY>-                     ;
3911                            UNIT=UIC,-                      ;
3912                            VERSION_MASK=<SYSGEN>
3913 ;
3914 ;          DEFAULT TERMINAL CLASS NAME PREFIX
3915 ;
3916            PARAMETER        ADDRESS=TTY$GW_CLASSNAM,-       ;
3917                            DEFAULT=<^A/TT/>,-      ;
3918                            MIN=<^A/AAAA/>,-        ;
3919                            MAX=<^A/ZZZZ/>,-        ;
3920                            NAME=TTY_CLASSNAME,-    ;
3921                            SIZE=WORD,-             ;
3922                            TYPE=<ASCII,TTY>,-      ;
3923                            UNIT=Ascii,-
3924                            VERSION_MASK=<SYSGEN>
3925 ;
3926 ;          DEFAULT SILO TIMEOUT VALUE FOR DMF32
3927 ;
3928            PARAMETER        ADDRESS=TTY$GB_SILOTIME,-       ;
3929                            DEFAULT=8,-     ;
3930                            MIN=0,-         ;
3931                            MAX=255,-       ;
3932                            NAME=TTY_SILOTIME,-     ;
```

```
                        SIZE=BYTE,-     ;
                        TYPE=<TTY>,-    ;
                        UNIT=Ms,-
                        VERSION_MASK=<SYSGEN>
;
; DISCONNECTED VIRTUAL TERMINAL TIMEOUT.
;
        PARAMETER       ADDRESS=TTY$GL_TIMEOUT,-        ;
                        DEFAULT=60*15,- ; 15 minute default
                        MIN=0,-         ;
                        NAME=TTY_TIMEOUT,-      ;
                        SIZE=LONG,-     ;
                        TYPE=<TTY,DYNAMIC>,-    ;
                        UNIT=Seconds,-
                        VERSION_MASK=<SYSGEN>
;
; AUTOBAUD RATE RECOGNITION CHARACTER
;
        PARAMETER       ADDRESS=TTY$GB_AUTOCHAR,-       ;
                        DEFAULT=7,-     ; Default is ^G (Bell)
                        MIN=0,-         ;
                        MAX=255,-       ;
                        NAME=TTY_AUTOCHAR,-     ;
                        SIZE=BYTE,-     ;
                        TYPE=<TTY,DYNAMIC>,-    ;
                        UNIT=Character,-
                        VERSION_MASK=<SYSGEN>
;
;       default port functions.
;
        PARAMETER       ADDRESS=TTY$GL_DEFPORT,-; DEFAULT PORT CHARACTERISTICS
                        DEFAULT=0,-
                        MIN=0,-                 ;
                        NAME=TTY_DEFPORT-       ;
                        SIZE=LONG,-             ;
                        TYPE=<TTY,SPECIAL>,-    ;
                        UNIT=Bit-Encoded,-
                        VERSION_MASK=<SYSGEN>
;
; END OF TERMINAL SYSTEM PARAMETERS
;
;
        .PAGE
        .SBTTL  RMS DEFAULT PARAMETERS
;
;  RMS DEFAULT PARAMETERS
;
        PARAMETER       ADDRESS=SYS$GB_DFMBC,-  ;
                        DEFAULT=16,-    ; DEFAULT MULTI-BLOCK COUNT
                        MIN=1,-                 ;
                        MAX=127,-               ;
                        NAME=RMS_DFMBC,-        ;
                        SIZE=BYTE,-             ;
                        TYPE=<RMS,DYNAMIC>,-    ;
                        UNIT=Blocks,-
                        VERSION_MASK=<SYSGEN>
;
```

```
3990 ; DEFAULT MULTI-BUFFER COUNT FOR SEQUENTIAL . DISK
3991 ;
3992         PARAMETER       ADDRESS=SYS$GB_DFMBFSDK,-       ;
3993                         DEFAULT=0,-             ;
3994                         MIN=0,-                 ;
3995                         MAX=127,-               ;
3996                         NAME=RMS_DFMBFSDK,-     ;
3997                         SIZE=BYTE,-             ;
3998                         TYPE=<RMS,DYNAMIC>,-    ;
3999                         UNIT=Blocks,-
4000                         VERSION_MASK=<SYSGEN>
4001 ;
4002 ; DEFAULT MULTI_BUFFER COUNT FOR MAGTAPE
4003 ;
4004         PARAMETER       ADDRESS=SYS$GB_DFMBFSMT,-       ;
4005                         DEFAULT=0,-             ;
4006                         MIN=0,-                 ;
4007                         MAX=127,-               ;
4008                         NAME=RMS_DFMBFSMT,-     ;
4009                         SIZE=BYTE,-             ;
4010                         TYPE=<RMS,DYNAMIC>,-    ;
4011                         UNIT=Blocks,-
4012                         VERSION_MASK=<SYSGEN>
4013 ;
4014 ;       DEFAULT MULTI-BUFFER COUNT FOR UNIT RECORD DEVICES.
4015 ;
4016         PARAMETER       ADDRESS=SYS$GB_DFMBFSUR,-       ;
4017                         DEFAULT=0,-             ;
4018                         MIN=0,-                 ;
4019                         MAX=127,-               ;
4020                         NAME=RMS_DFMBFSUR,-     ;
4021                         SIZE=BYTE,-             ;
4022                         TYPE=<RMS,DYNAMIC>,-    ;
4023                         UNIT=Buffers,-
4024                         VERSION_MASK=<SYSGEN>
4025 ;
4026 ; DEFAULT MULTI-BUFFER COUNT FOR RELATIVE FILES
4027 ;
4028         PARAMETER       ADDRESS=SYS$GB_DFMBFREL,-       ;
4029                         DEFAULT=0,-             ;
4030                         MIN=0,-                 ;
4031                         MAX=127,-               ;
4032                         NAME=RMS_DFMBFREL,-     ;
4033                         SIZE=BYTE,-             ;
4034                         TYPE=<RMS,DYNAMIC>,-    ;
4035                         UNIT=Buffers,-
4036                         VERSION_MASK=<SYSGEN>
4037 ;
4038 ; DEFAULT MULTI-BUFFER COUNT INDEXED FILES
4039 ;
4040         PARAMETER       ADDRESS=SYS$GB_DFMBFIDX,-       ;
4041                         DEFAULT=0,-             ;
4042                         MIN=0,-                 ;
4043                         MAX=127,-               ;
4044                         NAME=RMS_DFMBFIDX,-     ;
4045                         SIZE=BYTE,-             ;
4046                         TYPE=<RMS,DYNAMIC>,-    ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 75
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 4047                           UNIT=Buffers,-
 4048                           VERSION_MASK=<SYSGEN>
 4049 ;
 4050 ; DEFAULT MULTI-BUFFER COUNT HASHED
 4051 ;
 4052          PARAMETER        ADDRESS=SYS$GB_DFMBFHSH,-       ;
 4053                           DEFAULT=0,-             ;
 4054                           MIN=0,-                 ;
 4055                           MAX=127,-               ;
 4056                           NAME=RMS_DFMBFHSH,-     ;
 4057                           SIZE=BYTE,-             ;
 4058                           TYPE=<RMS,DYNAMIC>,-    ;
 4059                           UNIT=Buffers,-
 4060                           VERSION_MASK=<SYSGEN>
 4061
 4062 ;
 4063 ; Default rms Prologue
 4064 ;
 4065          PARAMETER        ADDRESS = SYS$GB_RMSPROLOG,-
 4066                           DEFAULT = 0,-           ; 0, 2 and 3 are valid, only
 4067                           MIN = 0,-
 4068                           MAX = 3,-
 4069                           NAME = RMS_PROLOGUE,-
 4070                           SIZE = BYTE,-
 4071                           TYPE = <RMS,DYNAMIC>,-
 4072                           UNIT = Prolog-Lvl,-
 4073                           VERSION_MASK=<SYSGEN>
 4074
 4075 ;
 4076 ; Default file extend quantity
 4077 ;
 4078
 4079          PARAMETER        ADDRESS = SYS$GW_RMSEXTEND,-
 4080                           DEFAULT = 0,-
 4081                           MIN = 0,-
 4082                           MAX = 65535,-
 4083                           NAME = RMS_EXTEND_SIZE,-
 4084                           SIZE = WORD,-
 4085                           TYPE = <RMS,DYNAMIC>,-
 4086                           UNIT = Blocks,-
 4087                           VERSION_MASK=<SYSGEN>
 4088 ;
 4089 ; Default file protection
 4090 ;
 4091
 4092          PARAMETER        ADDRESS = SYS$GW_FILEPROT,-
 4093                           DEFAULT = <^XFA00>,-
 4094                           MIN = 0,-
 4095                           MAX = <^XFFFF>,-
 4096                           NAME = RMS_FILEPROT,-
 4097                           SIZE = WORD,-
 4098                           TYPE = RMS,-
 4099                           UNIT = Prot-mask,-
 4100                           VERSION_MASK=<SYSGEN>
 4101
 4102 ;
 4103 ; Global buffer quota - This parameter determines the maximum number
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 76
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 4104 ; of global buffers that may be in use in the system at any one time.
 4105 ;
 4106         PARAMETER       ADDRESS=SYS$GW_GBLBUFQUO,-
 4107                         DEFAULT=1024,-
 4108                         MIN=0,-
 4109                         MAX=32767,-
 4110                         SIZE=WORD,-
 4111                         NAME=RMS_GBLBUFQUO,-
 4112                         TYPE=<RMS,DYNAMIC>,-
 4113                         UNIT=<Gbl bufs>,-
 4114                         VERSION_MASK=<SYSGEN>
 4115
 4116 ;
 4117 ; Network block count - This parameter determines the number
 4118 ; of blocks to use for RMS DAP network record-mode transfers.
 4119 ; This parameter determines the maximum record size that can be
 4120 ; sent over the network.
 4121 ;
 4122         PARAMETER       ADDRESS=SYS$GB_DFNBC,-
 4123                         DEFAULT=8,-
 4124                         MIN=1,-
 4125                         MAX=127,-
 4126                         SIZE=BYTE,-
 4127                         NAME=RMS_DFNBC,-
 4128                         TYPE=<RMS,DYNAMIC>,-
 4129                         UNIT=<Blocks>,-
 4130                         VERSION_MASK=<SYSGEN>
 4131
 4132         .IF     NOT_DEFINED GETSYISW
 4133
 4134         .IF     NOT_DEFINED,VERSION
 4135         .ALIGN  LONG
 4136         .ENDC
 4137 ;
 4138 ;  END OF RMS DEFAULT PARAMETERS
 4139 ;
 4140         .PAGE
 4141         .SBTTL  PROCESS QUOTA DEFAULTS AND MINIMA
 4142         .=.-4                           ;
 4143         DEFINE  PQL$AL_DEFAULT,-        ;
 4144                 VERSION_MASK=<SYSGEN>
 4145         .=.+4                           ;
 4146         .BLKL   PQL$_LENGTH             ;
 4147         .=.-4                           ;
 4148         DEFINE  PQL$AL_MIN,-            ;
 4149                 VERSION_MASK=<SYSGEN>
 4150         .=.+4                           ;
 4151         .BLKL   PQL$_LENGTH             ;
 4152         .=.-1                           ;
 4153         DEFINE  PQL$AB_FLAG,-           ;
 4154                 VERSION_MASK=<SYSGEN>
 4155         .=.+1                           ;
 4156         .BLKB   PQL$_LENGTH             ;
 4157
 4158         .ENDC   ; NOT_DEFINED GETSYISW
 4159
 4160 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 77
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

4161 ;       DEFINE PROCESS QUOTA DEFAULTS AND MINIMA
4162 ;
4163
4164         PQL     ASTLM,-                 ; AST LIMIT
4165                 DEFLT=24,-              ;
4166                 MINIM=4,-               ;
4167                 UNT=Ast,-               ;
4168                 PQL_VERSION_MASK=<SYSGEN>
4169
4170         PQL     BIOLM,-                 ; BUFFERED I/O LIMIT
4171                 DEFLT=18,-              ;
4172                 MINIM=4,-               ;
4173                 UNT=I/O,-                       ;
4174                 PQL_VERSION_MASK=<SYSGEN>
4175
4176         PQL     BYTLM,-                 ; BUFFERED I/O BYTE COUNT LIMIT
4177                 DEFLT=8192,-            ;
4178                 FLAG=DEDUCTIBLE,-       ;
4179                 MINIM=1024,-    ;
4180                 UNT=Bytes,-             ;
4181                 PQL_VERSION_MASK=<SYSGEN>
4182
4183         PQL     CPULM,-                 ; CPU TIME LIMIT
4184                 DEFLT=0,-               ;
4185                 FLAG=DEDUCTIBLE,-       ;
4186                 MINIM=0,-               ;
4187                 UNT=10Ms,-              ;
4188                 PQL_VERSION_MASK=<SYSGEN>
4189
4190         PQL     DIOLM,-                 ; DIRECT I/O LIMIT
4191                 DEFLT=18,-              ;
4192                 MINIM=4,-               ;
4193                 UNT=I/O,-                       ;
4194                 PQL_VERSION_MASK=<SYSGEN>
4195
4196         PQL     FILLM,-                 ; OPEN FILE LIMIT
4197                 DEFLT=16,-              ;
4198                 FLAG=DEDUCTIBLE,-       ;
4199                 MINIM=2,-               ;
4200                 UNT=Files,-             ;
4201                 PQL_VERSION_MASK=<SYSGEN>
4202
4203         PQL     PGFLQUOTA,-             ; PAGING FILE QUOTA
4204                 DEFLT=8192,-            ;
4205                 FLAG=DEDUCTIBLE,-       ;
4206                 MINIM=512,-             ;
4207                 UNT=Pages,-             ;
4208                 PQL_VERSION_MASK=<SYSGEN>
4209
4210         PQL     PRCLM,-                 ; SUB-PROCESS LIMIT
4211                 DEFLT=8,-               ;
4212                 FLAG=DEDUCTIBLE,-       ;
4213                 MINIM=0,-               ;
4214                 UNT=Processes,-
4215                 PQL_VERSION_MASK=<SYSGEN>
4216
4217         PQL     TQELM,-                 ; TIMER QUEUE ENTRY LIMIT
```

```
                        DEFLT=8,-               ;
                        FLAG=DEDUCTIBLE,-       ;
                        MINIM=0,-               ;
                        UNT=Timers,-
                        PQL_VERSION_MASK=<SYSGEN>

                PQL     WSDEFAULT,-             ; WORKING SET DEFAULT SIZE
                        DEFLT=100,-             ;
                        MINIM=60,-              ;
                        UNT=Pages,-             ;
                        DYNAMIC_FLAG=STATIC,-
                        PQL_VERSION_MASK=<SYSGEN>

;
; DEFAULT WORKING SET SIZE - SIZE OF SHELL WORKING SET
;
                .IF     NOT_DEFINED GETSYISW
                .IF     NDF,PRMSW               ;
                .IF     NOT_DEFINED     VERSION
SGN$GL_DFWSCNT==PQL$GDWSDEFAULT                 ; SYNONYM
                .IF_FALSE

                DEFINE  SGN$GL_DFWSCNT,-
                        VERSION_MASK=<SYSGEN>

                .ENDC                           ;
                .ENDC                           ;
                .ENDC   ; NOT_DEFINED GETSYISW
                PQL     WSQUOTA,-               ; WORKING SET QUOTA
                        DEFLT=200,-             ;
                        MINIM=60,-              ;
                        UNT=Pages,-             ;
                        PQL_VERSION_MASK=<SYSGEN>

                PQL     WSEXTENT,-              ; WORKING SET EXTENT
                        DEFLT=400,-             ;
                        MINIM=60,-              ;
                        UNT=Pages,-             ;
                        PQL_VERSION_MASK=<SYSGEN>

                PQL     ENQLM,-                 ; ENQUEUE LIMIT
                        DEFLT=30,-              ;
                        FLAG=DEDUCTIBLE,-       ;
                        MINIM=4,-               ;
                        UNT=Locks,-             ;
                        PQL_VERSION_MASK=<SYSGEN>

                PQL     JTQUOTA,-               ; JOB-WIDE LOGICAL NAME TABLE QUOTA
                        DEFLT=1024,-            ;
                        MINIM=0,-               ;
                        UNT=Bytes,-             ;
                        PQL_VERSION_MASK=<SYSGEN>

                .PAGE
                .SBTTL  FILE ACP CONFIGURATION DATA

;+
```

```
4275 ;
4276 ; File ACP configuration data. These parameters are used whenever an ACP is
4277 ; started up (or, in the case of per volume data, when a volume is mounted).
4278 ;
4279 ;-
4280
4281 ;
4282 ; Number of blocks in bitmap cache.
4283 ;
4284          PARAMETER       ADDRESS=ACP$GW_MAPCACHE,-       ;
4285                          DEFAULT=8,-     ;
4286                          MIN=1,-         ;
4287                          NAME=ACP_MAPCACHE,-     ;
4288                          SIZE=WORD,-     ;
4289                          TYPE=<ACP,DYNAMIC>,-    ;
4290                          UNIT=Pages,-    ;
4291                          VERSION_MASK=<SYSGEN>
4292 ;
4293 ; Number of blocks in file header cache.
4294 ;
4295          PARAMETER       ADDRESS=ACP$GW_HDRCACHE,-       ;
4296                          DEFAULT=32,-    ;
4297                          MIN=3,-         ;
4298                          NAME=ACP_HDRCACHE,-     ;
4299                          SIZE=WORD,-     ;
4300                          TYPE=<ACP,DYNAMIC>,-    ;
4301                          UNIT=Pages,-    ;
4302                          VERSION_MASK=<SYSGEN>
4303 ;
4304 ; Number of blocks in file system directory data block cache.
4305 ;
4306          PARAMETER       ADDRESS=ACP$GW_DIRCACHE,-       ;
4307                          DEFAULT=20,-    ;
4308                          MIN=2,-         ;
4309                          NAME=ACP_DIRCACHE,-     ;
4310                          SIZE=WORD,-     ;
4311                          TYPE=<ACP,DYNAMIC>,-    ;
4312                          UNIT=Pages,-    ;
4313                          VERSION_MASK=<SYSGEN>
4314 ;
4315 ; Number of pages in file system directory index cache.
4316 ;
4317          PARAMETER       ADDRESS=ACP$GW_DINDXCACHE,-     ;
4318                          DEFAULT=25,-    ;
4319                          MIN=2,-         ;
4320                          NAME=ACP_DINDXCACHE,-   ;
4321                          SIZE=WORD,-     ;
4322                          TYPE=<ACP,DYNAMIC>,-    ;
4323                          UNIT=Pages,-    ;
4324                          VERSION_MASK=<SYSGEN>
4325 ;
4326 ; ACP working set in pages (0 means maximal)
4327 ;
4328          PARAMETER       ADDRESS=ACP$GW_WORKSET,-        ;
4329                          DEFAULT=0,-     ;
4330                          MIN=0,-         ;
4331                          NAME=ACP_WORKSET,-      ;
```

```
                        SIZE=WORD,-     ;
                        TYPE=<ACP,DYNAMIC>,-    ;
                        UNIT=Pages,-    ;
                        VERSION_MASK=<SYSGEN>
;
; The following parameters are applied on a per volume basis.
;
;
; Number of cached index file slots
;
        PARAMETER       ADDRESS=ACP$GW_FIDCACHE,-       ;
                        DEFAULT=64,-    ;
                        MIN=0,-         ;
                        NAME=ACP_FIDCACHE,-     ;
                        SIZE=WORD,-     ;
                        TYPE=<ACP,DYNAMIC>,-    ;
                        UNIT=File-Ids,- ;
                        VERSION_MASK=<SYSGEN>
;
; Number of cached disk extents
;
        PARAMETER       ADDRESS=ACP$GW_EXTCACHE,-       ;
                        DEFAULT=64,-    ;
                        MIN=0,-         ;
                        NAME=ACP_EXTCACHE,-     ;
                        SIZE=WORD,-     ;
                        TYPE=<ACP,DYNAMIC>,-    ;
                        UNIT=Extents,-  ;
                        VERSION_MASK=<SYSGEN>
;
; Maximum fraction of disk to cache in tenths of percent
;
        PARAMETER       ADDRESS=ACP$GW_EXTLIMIT,-       ;
                        DEFAULT=100,-   ;
                        MIN=0,-         ;
                        MAX=1000,-      ;
                        NAME=ACP_EXTLIMIT,-     ;
                        SIZE=WORD,-     ;
                        TYPE=<ACP,DYNAMIC>,-    ;
                        UNIT=<Percent/10>,-     ;
                        VERSION_MASK=<SYSGEN>

;
; Number of quota file entries to cache
;
        PARAMETER       ADDRESS=ACP$GW_QUOCACHE,-
                        DEFAULT=64,-
                        MIN=0,-
                        MAX=-1,-
                        NAME=ACP_QUOCACHE,-
                        SIZE=WORD,-
                        TYPE=<ACP,DYNAMIC>,-
                        UNIT=Users,-
                        VERSION_MASK=<SYSGEN>
;
; Default access for system volumes
;
```

```
4389          PARAMETER       ADDRESS=ACP$GW_SYSACC,- ;
4390                          DEFAULT=8,-     ;
4391                          MIN=0,-         ;
4392                          NAME=ACP_SYSACC,-       ;
4393                          SIZE=WORD,-             ;
4394                          TYPE=<ACP,DYNAMIC>,-    ;
4395                          UNIT=Directories,-
4396                          VERSION_MASK=<SYSGEN>
4397 ;
4398 ; Maximum number of blocks to read at once for directories.
4399 ;
4400          PARAMETER       ADDRESS=ACP$GB_MAXREAD,-        ;
4401                          DEFAULT=32,-    ;
4402                          MIN=1,-         ;
4403                          MAX=64,-        ;
4404                          NAME=ACP_MAXREAD,-      ;
4405                          SIZE=BYTE,-     ;
4406                          TYPE=<ACP,DYNAMIC>,-    ;
4407                          UNIT=Blocks,-   ;
4408                          VERSION_MASK=<SYSGEN>
4409 ;
4410 ; Default window size for system volumes.
4411 ;
4412          PARAMETER       ADDRESS=ACP$GB_WINDOW,- ;
4413                          DEFAULT=7,-     ;
4414                          MIN=1,-         ;
4415                          NAME=ACP_WINDOW,-       ;
4416                          SIZE=BYTE,-     ;
4417                          TYPE=<ACP,DYNAMIC>,-    ;
4418                          UNIT=Pointers,- ;
4419                          VERSION_MASK=<SYSGEN>
4420 ;
4421 ; Deferred cache writeback enable.
4422 ;
4423          PARAMETER       ADDRESS=ACP$GB_WRITBACK,-
4424                          DEFAULT=1,-
4425                          MIN=0,-
4426                          MAX=1,-
4427                          NAME=ACP_WRITEBACK,-
4428                          SIZE=BYTE,-
4429                          TYPE=<ACP,DYNAMIC>,-
4430                          UNIT=Boolean,-
4431                          VERSION_MASK=<SYSGEN>
4432 ;
4433 ; ACP datacheck enable flags.
4434 ;
4435          PARAMETER       ADDRESS=ACP$GB_DATACHK,-
4436                          DEFAULT=2,-
4437                          MIN=0,-
4438                          MAX=3,-
4439                          NAME=ACP_DATACHECK,-
4440                          SIZE=BYTE,-
4441                          TYPE=<ACP,DYNAMIC>,-
4442                          UNIT=Bit-mask,-
4443                          VERSION_MASK=<SYSGEN>
4444 ;
4445 ; Containing the following flags:
```

```
4446 ;
4447          .IF     NOT_DEFINED     VERSION
4448          .IF     NDF,PRMSW               ; Only for exec version of sysparam
4449          $GBLINI GLOBAL
4450          $VIELD  ACP,0,<-
4451                  <READCHK>-              ; do datachecks on reads
4452                  <WRITECHK>-             ; do datachecks on writes
4453                  >
4454          .ENDC
4455          .IF_FALSE
4456
4457          DEFINE  ACP$V_READCHK,-
4458                  VERSION_MASK=<SYSGEN>
4459
4460          DEFINE  ACP$V_WRITECHK,-
4461                  VERSION_MASK=<SYSGEN>
4462
4463          .ENDC                           ;
4464 ;
4465 ; The following parameters apply per ACP.
4466 ;
4467 ;
4468 ; ACP base priority
4469 ;
4470          PARAMETER       ADDRESS=ACP$GB_BASEPRIO,-       ;
4471                          DEFAULT=8,-     ;
4472                          MIN=4,-         ;
4473                          MAX=31,-        ;
4474                          NAME=ACP_BASEPRIO,-     ;
4475                          SIZE=BYTE,-     ;
4476                          TYPE=<ACP,DYNAMIC>,-    ;
4477                          UNIT=Priority,-
4478                          VERSION_MASK=<SYSGEN>
4479 ;
4480 ; ACP Swap flags
4481 ;
4482          PARAMETER       ADDRESS=ACP$GB_SWAPFLGS,-       ;
4483                          DEFAULT=<^B1111>,-      ;.
4484                          MIN=0,-         ;
4485                          MAX=15,-        ;
4486                          NAME=ACP_SWAPFLGS,-     ;
4487                          SIZE=BYTE,-     ;
4488                          TYPE=<ACP,DYNAMIC>-     ;
4489                          UNIT=Bit-mask,-
4490                          VERSION_MASK=<SYSGEN>
4491                                          ; swappable, as follows:
4492          .IF     NOT_DEFINED     VERSION
4493          .IF     NDF,PRMSW               ; Only for exec version of sysparam
4494          $GBLINI GLOBAL
4495          $VIELD  ACP,0,<-
4496                  <SWAPSYS>-              ; /SYSTEM
4497                  <SWAPGRP>-              ; /GROUP
4498                  <SWAPPRV>-              ; other (private mount)
4499                  <SWAPMAG>-              ; magtape
4500                  >
4501          .ENDC
4502          .IF_FALSE
```

```
4503
4504          DEFINE  ACP$V_SWAPSYS,-
4505                  VERSION_MASK=<SYSGEN>
4506
4507          DEFINE  ACP$V_SWAPGRP,-
4508                  VERSION_MASK=<SYSGEN>
4509
4510          DEFINE  ACP$V_SWAPPRV,-
4511                  VERSION_MASK=<SYSGEN>
4512
4513          DEFINE  ACP$V_SWAPMAG,-
4514                  VERSION_MASK=<SYSGEN>
4515
4516          .ENDC                           ;
4517 ;
4518 ; XQP memory resident control flag
4519 ;
4520          PARAMETER       ADDRESS=EXE$GL_STATIC_FLAGS,-
4521                          DEFAULT=1,-
4522                          MIN=0,-
4523                          MAX=1,-
4524                          NAME=ACP_XQP_RES,-
4525                          BIT=EXE$V_XQP_RESIDENT,-
4526                          TYPE=<ACP,STATIC>,-
4527                          UNIT=Boolean,-
4528                          VERSION_MASK=<SYSGEN>
4529 ;
4530 ; System disk rebuild flag.
4531 ;
4532          PARAMETER       ADDRESS=EXE$GL_STATIC_FLAGS,-
4533                          DEFAULT=1,-
4534                          MIN=0,-
4535                          MAX=1,-
4536                          NAME=ACP_REBLDSYSD,-
4537                          BIT=EXE$V_REBLDSYSD,-
4538                          TYPE=<ACP,STATIC>,-
4539                          UNIT=Boolean,-
4540                          VERSION_MASK=<SYSGEN>
4541
4542 ;
4543 ; Stand-alone application boot flag.
4544 ;
4545          PARAMETER       ADDRESS=EXE$GL_STATIC_FLAGS,-
4546                          DEFAULT=0,-
4547                          MIN=0,-
4548                          MAX=1,-
4549                          NAME=SA_APP,-
4550                          BIT=EXE$V_SA_APP,-
4551                          TYPE=<SPECIAL,STATIC>,-
4552                          UNIT=Boolean,-
4553                          VERSION_MASK=<SYSGEN>
4554
4555          .IF     NOT_DEFINED VERSION
4556          .IIF    NOT_DEFINED GETSYISW,   .ALIGN  LONG
4557          .ENDC
4558
4559          .PAGE
```

```
4560          .SBTTL  Job Controller Parameters
4561 ;
4562 ; Default Priority for Job Initiations
4563 ;
4564          PARAMETER       SYS$GB_DEFPRI,- ;
4565                          DEFAULT=4,-     ;
4566                          MIN=1,-         ;
4567                          MAX=31,-        ;
4568                          NAME=DEFPRI,-   ;
4569                          SIZE=BYTE,-     ;
4570                          TYPE=<SYS,JBC,DYNAMIC>,-        ;
4571                          UNIT=Priority,-
4572                          VERSION_MASK=<SYSGEN>
4573 ;
4574 ; Limit for interactive Jobs
4575 ;
4576          PARAMETER       ADDRESS=SYS$GW_IJOBLIM,-        ;
4577                          DEFAULT=64,-    ;
4578                          MIN=1,-         ;
4579                          MAX=1024,-      ;
4580                          NAME=IJOBLIM,-  ;
4581                          SIZE=WORD,-     ;
4582                          TYPE=<JBC,DYNAMIC>,-    ;
4583                          UNIT=Jobs,-
4584                          VERSION_MASK=<SYSGEN>
4585 ;
4586 ; Limit for batch Jobs
4587 ;
4588          PARAMETER       ADDRESS=SYS$GW_BJOBLIM,-        ;
4589                          DEFAULT=16,-    ;
4590                          MIN=0,-         ;
4591                          MAX=1024,-      ;
4592                          NAME=BJOBLIM,-  ;
4593                          SIZE=WORD,-     ;
4594                          TYPE=<JBC,DYNAMIC>,-    ;
4595                          UNIT=Jobs,-
4596                          VERSION_MASK=<SYSGEN>
4597
4598 ;
4599 ; Limit for network Jobs
4600 ;
4601          PARAMETER       ADDRESS=SYS$GW_NJOBLIM,-        ;
4602                          DEFAULT=16,-    ;
4603                          MIN=0,-         ;
4604                          MAX=1024,-      ;
4605                          NAME=NJOBLIM,-  ;
4606                          SIZE=WORD,-     ;
4607                          TYPE=<JBC,DYNAMIC>,-    ;
4608                          UNIT=Jobs,-
4609                          VERSION_MASK=<SYSGEN>
4610 ;
4611 ; Limit for Remote Terminal Jobs
4612 ;
4613          PARAMETER       ADDRESS=SYS$GW_RJOBLIM,-        ;
4614                          DEFAULT=16,-    ;
4615                          MIN=0,-         ;
4616                          MAX=65535,-     ;
```

```
4617                            NAME=RJOBLIM,-  ;
4618                            SIZE=WORD,-     ;
4619                            TYPE=<JBC,DYNAMIC>,-    ;
4620                            UNIT=Jobs,-
4621                            VERSION_MASK=<SYSGEN>
4622 ;
4623 ;       DEFQUEPRI - Default Queue Priority
4624 ;
4625            PARAMETER        ADDRESS=SYS$GB_DEFQUEPRI,-
4626                             DEFAULT=100,-
4627                             MAX=255,-
4628                             MIN=0,-
4629                             NAME=DEFQUEPRI,-
4630                             TYPE=<JBC,DYNAMIC>,-
4631                             UNIT=Priority,-
4632                             VERSION_MASK=<SYSGEN>
4633 ;
4634 ;       MAXQUEPRI - Maximum Queue Priority
4635 ;
4636            PARAMETER        ADDRESS=SYS$GB_MAXQUEPRI,-
4637                             DEFAULT=100,-
4638                             MAX=255,-
4639                             MIN=0,-
4640                             NAME=MAXQUEPRI,-
4641                             TYPE=<JBC,DYNAMIC>,-
4642                             UNIT=Priority,-
4643                             VERSION_MASK=<SYSGEN>
4644
4645            .PAGE
4646            .SBTTL  Login Security Parameters
4647 ;
4648 ; Number of seconds that a dialup user has in which to enter the system
4649 ; password before LOGINOUT goes away
4650 ;
4651            PARAMETER        ADDRESS=SYS$GB_PWD_TMO,-
4652                             DEFAULT=30,-
4653                             MIN=0,-
4654                             MAX=255,-
4655                             NAME=LGI_PWD_TMO,-
4656                             SIZE=BYTE,-
4657                             TYPE=<DYNAMIC,LGI>,-
4658                             UNIT=Seconds,-
4659                             VERSION_MASK=<SYSGEN>
4660
4661 ;
4662 ; Number of retries an interactive user has before the process goes away
4663 ;
4664            PARAMETER        ADDRESS=SYS$GB_RETRY_LIM,-
4665                             DEFAULT=3,-
4666                             MIN=0,-
4667                             MAX=255,-
4668                             NAME=LGI_RETRY_LIM,-
4669                             SIZE=BYTE,-
4670                             TYPE=<DYNAMIC,LGI>,-
4671                             UNIT=Tries,-
4672                             VERSION_MASK=<SYSGEN>
4673
```

```
 4674 ;
 4675 ; Number of seconds that a user has in which to attempt another login
 4676 ; before the process goes away
 4677 ;
 4678         PARAMETER       ADDRESS=SYS$GB_RETRY_TMO,-
 4679                         DEFAULT=20,-
 4680                         MIN=0,-
 4681                         MAX=255,-
 4682                         NAME=LGI_RETRY_TMO,-
 4683                         SIZE=BYTE,-
 4684                         TYPE=<DYNAMIC,LGI>,-
 4685                         UNIT=Seconds,-
 4686                         VERSION_MASK=<SYSGEN>
 4687
 4688 ;
 4689 ; Number of consecutive login failures before LOGINOUT begins evasive action
 4690 ;
 4691         PARAMETER       ADDRESS=SYS$GB_BRK_LIM,-
 4692                         DEFAULT=5,-
 4693                         MIN=0,-
 4694                         MAX=255,-
 4695                         NAME=LGI_BRK_LIM,-
 4696                         SIZE=BYTE,-
 4697                         TYPE=<DYNAMIC,LGI>,-
 4698                         UNIT=Failures,-
 4699                         VERSION_MASK=<SYSGEN>
 4700
 4701 ;
 4702 ; Number of seconds that a suspect must be free of login failures before it is
 4703 ; taken off the suspect list
 4704 ;
 4705         PARAMETER       ADDRESS=SYS$GL_BRK_TMO,-
 4706                         DEFAULT=300,-
 4707                         MIN=0,-
 4708                         MAX=-1,-
 4709                         NAME=LGI_BRK_TMO,-
 4710                         SIZE=LONG,-
 4711                         TYPE=<DYNAMIC,LGI>,-
 4712                         UNIT=Seconds,-
 4713                         VERSION_MASK=<SYSGEN>
 4714
 4715 ;
 4716 ; Number of seconds that LOGINOUT should practice evasive action on an intruder
 4717 ;
 4718         PARAMETER       ADDRESS=SYS$GL_HID_TIM,-
 4719                         DEFAULT=300,-
 4720                         MIN=0,-
 4721                         MAX=-1,-
 4722                         NAME=LGI_HID_TIM,-
 4723                         SIZE=LONG,-
 4724                         TYPE=<DYNAMIC,LGI>,-
 4725                         UNIT=Seconds,-
 4726                         VERSION_MASK=<SYSGEN>
 4727
 4728         .PAGE
 4729         .SBTTL  Cluster Parameters
 4730
```

```
4731 ;
4732 ;       VAXCLUSTER - Controls loading of cluster code
4733 ;               0: Never load
4734 ;               1: Load if SCSLOA is being loaded
4735 ;               2: Always load (and also load SCSLOA)
4736 ;
4737         PARAMETER       ADDRESS=CLU$GB_VAXCLUSTER, -    ;
4738                         DEFAULT=1,-     ;
4739                         MAX=2,-         ;
4740                         MIN=0,-         ;
4741                         NAME=VAXCLUSTER,-       ;
4742                         SIZE=BYTE,-     ;
4743                         TYPE=<CLUSTER>,-        ;
4744                         UNIT=Coded-value,-
4745                         VERSION_MASK=<SYSGEN>
4746
4747 ;
4748 ; Maximum number of votes that are expected to be in the cluster.
4749 ; Also known as the universe of votes.
4750 ;
4751         PARAMETER       ADDRESS=CLU$GW_EXP_VOTES,-
4752                         DEFAULT=1,-
4753                         MIN=1,-
4754                         MAX=127,-
4755                         NAME=EXPECTED_VOTES,-
4756                         SIZE=WORD,-
4757                         TYPE=<CLUSTER>,-
4758                         UNIT=Votes,-
4759                         VERSION_MASK=<SYSGEN>
4760
4761 ;
4762 ; Number of votes this system contributes to quorum
4763 ;
4764         PARAMETER       ADDRESS=CLU$GW_VOTES,-
4765                         DEFAULT=1,-
4766                         MIN=0,-
4767                         MAX=127,-
4768                         NAME=VOTES,-
4769                         SIZE=WORD,-
4770                         TYPE=<CLUSTER>,-
4771                         UNIT=Votes,-
4772                         VERSION_MASK=<SYSGEN>
4773
4774 ;
4775 ; Interval during which to attempt reconnection to a remote system
4776 ;
4777         PARAMETER       ADDRESS=CLU$GW_RECNXINT,-
4778                         DEFAULT=20,-
4779                         MIN=1,-
4780                         MAX=32767,-
4781                         NAME=RECNXINTERVAL,-
4782                         SIZE=WORD,-
4783                         TYPE=<CLUSTER,DYNAMIC>,-
4784                         UNIT=Seconds,-
4785                         VERSION_MASK=<SYSGEN>
4786
4787 ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 88
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

 4788 ; The cluster quorum disk name
 4789 ;
 4790         PARAMETER       ADDRESS=CLU$GB_QDISK,-
 4791                         DEFAULT=<^A/    />,-
 4792                         MIN=<^A/    />,-
 4793                         MAX=<^A/ZZZZ/>,-
 4794                         NAME=DISK_QUORUM,-
 4795                         SIZE=OCTA,-
 4796                         TYPE=<ASCII,CLUSTER>,-
 4797                         UNIT=Ascii,-
 4798                         VERSION_MASK=<SYSGEN>
 4799 ;
 4800 ; Number of votes contributed by quorum disk
 4801 ;
 4802         PARAMETER       ADDRESS=CLU$GW_QDSKVOTES,-
 4803                         DEFAULT=1,-
 4804                         MIN=0,-
 4805                         MAX=127,-
 4806                         NAME=QDSKVOTES,-
 4807                         SIZE=WORD,-
 4808                         TYPE=<CLUSTER>,-
 4809                         UNIT=Votes,-
 4810                         VERSION_MASK=<SYSGEN>
 4811
 4812 ;
 4813 ; Disk Quorum Inverval
 4814 ;
 4815         PARAMETER       ADDRESS=CLU$GW_QDSKINTERVAL,-
 4816                         DEFAULT=10,-
 4817                         MIN=1,-
 4818                         MAX=32767,-
 4819                         NAME=QDSKINTERVAL,-
 4820                         SIZE=WORD,-
 4821                         TYPE=<CLUSTER>,-
 4822                         UNIT=Seconds,-
 4823                         VERSION_MASK=<SYSGEN>
 4824
 4825 ;
 4826 ; Define a parameter which determines the disk allocation class for
 4827 ; this system.  The disk device allocation class is used to derive a common
 4828 ; lock resource name for multiple access paths to the same disk device.
 4829 ;
 4830         PARAMETER       ADDRESS=CLU$GL_ALLOCLS, -
 4831                         NAME=ALLOCLASS, -
 4832                         TYPE=<CLUSTER>, -
 4833                         DEFAULT=0, -
 4834                         MIN=0, -
 4835                         MAX=255, -
 4836                         UNIT=Pure-number,-
 4837                         VERSION_MASK=<SYSGEN>
 4838
 4839 ;
 4840 ; Lock manager directory system weight.  Determines portion of lock
 4841 ; manger directory entires which will be handled by this system.
 4842 ;
 4843         PARAMETER       ADDRESS=CLU$GW_LCKDIRWT,-
 4844                         DEFAULT=0,-
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 89
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

                        MIN=0,-
                        MAX=255,-
                        NAME=LOCKDIRWT,-
                        SIZE=WORD,-
                        TYPE=<CLUSTER>,-
                        UNIT=Pure-number,-
                        VERSION_MASK=<SYSGEN>

        .IF     NOT_DEFINED VERSION
        .IF     NOT_DEFINED GETSYISW
;
; Cluster boolean flags
;
        .ALIGN  LONG                    ;
        .IF     NDF,PRMSW               ;
CLU$GL_SGN_FLAGS::                      ; Static SYSGEN flags
        .ENDC   ; NDF,PRMSW
        .LONG   0                       ; init to <1@CLU$V_xxx>!<1@CLU$V_xxx>
        .ENDC   ; NOT_DEFINED GETSYISW

        $VIELD  CLU,0,<-                ; Definition for CLU$GL_SGN_FLAGS
                NISCS_LOAD_PEA0,-       ; Load PEA0 for NISCS
                NISCS_CONV_BOOT,-       ; Allow remote conversational boot
        >
        .IF_FALSE

        DEFINE  CLU$GL_SGN_FLAGS,-
                VERSION_MASK=<SYSGEN>

        DEFINE  CLU$V_NISCS_LOAD_PEA0,-
                VERSION_MASK=<SYSGEN>

        DEFINE  CLU$V_CONV_BOOT,-
                VERSION_MASK=<SYSGEN>

        .ENDC   ; Version


;
; Local Area VAXcluster parameters
;
; Allow remotely booting node to do conversational boots
;
        PARAMETER       ADDRESS=CLU$GL_SGN_FLAGS, -
                        NAME=NISCS_CONV_BOOT, -
                        TYPE=<CLUSTER>, -
                        BIT=CLU$V_NISCS_CONV_BOOT, -
                        DEFAULT=0,-     ;
                        MIN=0,-         ;
                        MAX=1,-         ;
                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>
;
; Load PEA0 for NI-SCS
;
        PARAMETER       ADDRESS=CLU$GL_SGN_FLAGS, -
                        NAME=NISCS_LOAD_PEA0, -
```

```
                        TYPE=<CLUSTER>, -
                        BIT=CLU$V_NISCS_LOAD_PEA0, -
                        DEFAULT=0,-     ;
                        MIN=0,-         ;
                        MAX=1,-         ;
                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>
;
; Flags and other bits for port service.  Currently a set of
; bit flags, but can be expanded
;
        PARAMETER       ADDRESS=CLU$GL_NISCS_PORT_SERV, -
                        NAME=NISCS_PORT_SERV, -
                        TYPE=<CLUSTER>, -
                        DEFAULT=0,-     ;
                        MIN=0,-         ;
                        MAX=3,-         ;
                        UNIT=Bit-encoded,-
                        VERSION_MASK=<SYSGEN>

;
; Load MSCP server
;
        PARAMETER       ADDRESS=CLU$GL_MSCP_LOAD, -
                        NAME=MSCP_LOAD, -
                        TYPE=<CLUSTER>, -
                        DEFAULT=0,-
                        MIN=0,-
                        MAX=1,-
                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>

;
; MSCP-serve all disks at boot time
;
        PARAMETER       ADDRESS=CLU$GL_MSCP_SERVE_ALL, -
                        NAME=MSCP_SERVE_ALL, -
                        TYPE=<CLUSTER>, -
                        DEFAULT=0,-
                        MIN=0,-
                        MAX=2,-
                        UNIT=Coded-value,-
                        VERSION_MASK=<SYSGEN>

;
; Amount of non-paged pool to allocate for the servers exclusive use
;
        PARAMETER       ADDRESS=CLU$GL_MSCP_BUFFER, -
                        NAME=MSCP_BUFFER, -
                        TYPE=<CLUSTER>, -
                        DEFAULT=128,-
                        MIN=16,-
                        UNIT=Coded-value,-
                        VERSION_MASK=<SYSGEN>
;
; Number of send credits to be extended by the MSCP server for each
; granted connection.
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 91
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

4959 ;
4960          PARAMETER       ADDRESS=CLU$GL_MSCP_CREDITS, -
4961                          NAME=MSCP_CREDITS, -
4962                          TYPE=<CLUSTER>, -
4963                          DEFAULT=4,-
4964                          MIN=2,-
4965                          MAX=8,-
4966                          UNIT=Coded-value,-
4967                          VERSION_MASK=<SYSGEN>
4968
4969 ;
4970 ; Define a parameter which tells us whether or not this system is tailored
4971 ; (i.e., has a library disk).  This is the right way to determine tailoring
4972 ; now that many different system disks can be tailored.
4973 ;
4974          PARAMETER       ADDRESS=SGN$GB_TAILORED,-
4975                          DEFAULT=0,-
4976                          MAX=1,-
4977                          MIN=0,-
4978                          NAME=TAILORED,-
4979                          SIZE=BYTE,-
4980                          TYPE=<SYS>,-
4981                          UNIT=Boolean,-
4982                          VERSION_MASK=<SYSGEN>
4983
4984          .IF     NOT_DEFINED VERSION
4985          .IF     NOT_DEFINED GETSYISW
4986 ;
4987 ; WORK STATION FLAGS.
4988 ;
4989          .ALIGN  LONG                    ;
4990          .IF     NDF,PRMSW               ;
4991 EXE$GL_WSFLAGS::                         ; Work station SYSGEN flags
4992          .ENDC   ; NDF,PRMSW
4993          .LONG   0;
4994          .ENDC   ; NOT_DEFINED GETSYISW
4995
4996          $VIELD  EXE,0,<-                ; DEFINITION FOR EXE$GL_DYNAMIC_FLAGS
4997                  <OPA0>-                 ; Reserve a window for OPA0
4998          >
4999          .IF_FALSE
5000
5001          DEFINE  EXE$GL_WSFLAGS,-
5002                  VERSION_MASK=<SYSGEN>
5003
5004          DEFINE  EXE$V_OPA0,-
5005                  VERSION_MASK=<SYSGEN>
5006
5007          .ENDC   ; Version
5008
5009
5010 ;
5011 ;        WS_OPA0 -       If set reserve the first 24 scan lines for an OPA0 window
5012 ;
5013          PARAMETER       ADDRESS=exe$gl_wsflags,-;
5014                          DEFAULT=0,-     ;
5015                          MAX=1,-         ;
```

```
                        MIN=0,-         ;
                        NAME=WS_OPA0,-  ;
                        BIT=EXE$V_OPA0,-;
                        TYPE=<DYNAMIC,SYS>,-;
                        UNIT=Boolean,-
                        VERSION_MASK=<SYSGEN>


;
; Define eight parameters which are used to pass information to the system
; startup procedure (STARTUP.COM).
;
        PARAMETER       ADDRESS=SGN$GB_STARTUP_P1,-
                        DEFAULT=<^A/    />,-
                        MIN=<^A/    />,-
                        MAX=<^A/zzzz/>,-
                        NAME=STARTUP_P1,-
                        SIZE=LONG,-
                        TYPE=<ASCII,SYS>,-
                        UNIT=Ascii,-
                        VERSION_MASK=<SYSGEN>
        PARAMETER       ADDRESS=SGN$GB_STARTUP_P2,-
                        DEFAULT=<^A/    />,-
                        MIN=<^A/    />,-
                        MAX=<^A/zzzz/>,-
                        NAME=STARTUP_P2,-
                        SIZE=LONG,-
                        TYPE=<ASCII,SYS>,-
                        UNIT=Ascii,-
                        VERSION_MASK=<SYSGEN>
        PARAMETER       ADDRESS=SGN$GB_STARTUP_P3,-
                        DEFAULT=<^A/    />,-
                        MIN=<^A/    />,-
                        MAX=<^A/zzzz/>,-
                        NAME=STARTUP_P3,-
                        SIZE=LONG,-
                        TYPE=<ASCII,SYS>,-
                        UNIT=Ascii,-
                        VERSION_MASK=<SYSGEN>
        PARAMETER       ADDRESS=SGN$GB_STARTUP_P4,-
                        DEFAULT=<^A/    />,-
                        MIN=<^A/    />,-
                        MAX=<^A/zzzz/>,-
                        NAME=STARTUP_P4,-
                        SIZE=LONG,-
                        TYPE=<ASCII,SYS>,-
                        UNIT=Ascii,-
                        VERSION_MASK=<SYSGEN>
        PARAMETER       ADDRESS=SGN$GB_STARTUP_P5,-
                        DEFAULT=<^A/    />,-
                        MIN=<^A/    />,-
                        MAX=<^A/zzzz/>,-
                        NAME=STARTUP_P5,-
                        SIZE=LONG,-
                        TYPE=<ASCII,SYS>,-
                        UNIT=Ascii,-
                        VERSION_MASK=<SYSGEN>
```

```
5073          PARAMETER       ADDRESS=SGN$GB_STARTUP_P6,-
5074                          DEFAULT=<^A/    />,-
5075                          MIN=<^A/    />,-
5076                          MAX=<^A/zzzz/>,-
5077                          NAME=STARTUP_P6,-
5078                          SIZE=LONG,-
5079                          TYPE=<ASCII,SYS>,-
5080                          UNIT=Ascii,-
5081                          VERSION_MASK=<SYSGEN>
5082          PARAMETER       ADDRESS=SGN$GB_STARTUP_P7,-
5083                          DEFAULT=<^A/    />,-
5084                          MIN=<^A/    />,-
5085                          MAX=<^A/zzzz/>,-
5086                          NAME=STARTUP_P7,-
5087                          SIZE=LONG,-
5088                          TYPE=<ASCII,SYS>,-
5089                          UNIT=Ascii,-
5090                          VERSION_MASK=<SYSGEN>
5091          PARAMETER       ADDRESS=SGN$GB_STARTUP_P8,-
5092                          DEFAULT=<^A/    />,-
5093                          MIN=<^A/    />,-
5094                          MAX=<^A/zzzz/>,-
5095                          NAME=STARTUP_P8,-
5096                          SIZE=LONG,-
5097                          TYPE=<ASCII,SYS>,-
5098                          UNIT=Ascii,-
5099                          VERSION_MASK=<SYSGEN>
5100
5101 ;
5102 ;       ENABLE PAGING OF SYSTEM CODE
5103 ;
5104          PARAMETER       ADDRESS=EXE$GL_S0_PAGING,-;
5105                          DEFAULT=0,-     ;
5106                          MIN=0,-         ;
5107                          NAME=S0_PAGING,-;
5108                          TYPE=<SPECIAL>,-        ;
5109                          UNIT=Bit-mask,-
5110                          VERSION_MASK=<SYSGEN>
5111 ;
5112 ;       SET VALUES FOR POOLCHECK CODE IN MEMORYALC
5113 ;       (Field test default line is DEFAULT=<^X6164001F>)
5114 ;
5115          PARAMETER       ADDRESS=EXE$GL_POOLCHECK,-      ;
5116                          DEFAULT=0,-                     ;
5117                          MIN=0,-                         ;
5118                          NAME=POOLCHECK,-                ;
5119                          SIZE=LONG,-                     ;
5120                          TYPE=<DYNAMIC,SPECIAL>,-        ;
5121                          UNIT=Bit-encoded,-
5122                          VERSION_MASK=<SYSGEN>
5123 ;
5124 ;       SET SIZE OF PSEUDO DEVICE ON BOOT (AND TAPE BOOT FLAG)
5125 ;
5126          PARAMETER       ADDRESS=SGN$GL_PSEUDOLOA,-      ;
5127                          DEFAULT=0,-                     ;
5128                          MIN=0,-                         ;
5129                          NAME=PSEUDOLOA,-                ;
```

```
                        SIZE=LONG,-                     ;
                        TYPE=<SPECIAL>,-                ;
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>
;
        .PAGE
        .SBTTL  CPU Scheduling Control Flags and Cells

        PARAMETER       ADDRESS=SCH$GL_CTLFLAGS,-;
                        DEFAULT=0,-     ; All bits zero selects pre-V5 CPU schedulin
                        MIN=0,-         ;
                        NAME=SCH_CTLFLAGS,-;
                        TYPE=<DYNAMIC,SPECIAL>,-;
                        UNIT=Bit-mask,-
                        VERSION_MASK=<SYSGEN>


;
; Note that the defaults for the following 3 parameters establish a priority
; structure that appears as:
;
;       priority        31
;                        .
;                        .      Real time
;                        .
;                       16
;-------------------------------------------------------------------------------
---
;                       15
;                        .
;                        .      Normal, pre-emptive resume
;                        .      (assuming NO_QEND_PREEMPT = 1)
;                        8
;-------------------------------------------------------------------------------
---
;                        7
;                        .
;                        .      Class-scheduled processes
;                        .      (assuming class scheduling is enabled)
;                        1
;-------------------------------------------------------------------------------
---
;                        0      NULL or SOAK processes
;                               (assuming NO_PIXSCAN = 1 OR PIXSCAN_LOWPRI = 1)

;
;
; If class scheduling is enabled, then the following two parameters
; delineate the priority range in which class-scheduled processes will
; execute.
;
;       MINCLASSPRI <= process priority <= MAXCLASSPRI
;
        PARAMETER       ADDRESS=SCH$GB_MINCLASSPRI,-    ;
                        DEFAULT=1,-     ;
                        MIN=0,-         ;
                        MAX=15,-        ;
                        NAME=MINCLASSPRI,-;
                        SIZE=BYTE,-     ;
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 95
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

                        TYPE=<DYNAMIC,SPECIAL>,-;
                        UNIT=Priority,-
                        VERSION_MASK=<SYSGEN>

        PARAMETER       ADDRESS=SCH$GB_MAXCLASSPRI,-    ;
                        DEFAULT=7,-     ;
                        MIN=0,-         ;
                        MAX=15,-        ;
                        NAME=MAXCLASSPRI,-;
                        SIZE=BYTE,-     ;
                        TYPE=<DYNAMIC,SPECIAL>,-;
                        UNIT=Priority,-
                        VERSION_MASK=<SYSGEN>

;
; If the scheduling control flag NO_QEND_PREEMPT is set, then the following
; parameter divides the non-realtime priority range into 2 classes:
;
;        > MINPRPRI - NO_QEND_PREEMPT = 1 enables priority-resume scheduling
;       <= MINPRPRI - NO_QEND_PREEMPT has no effect, i.e., round-robin scheduling is
;
        PARAMETER       ADDRESS=SCH$GB_MINPRPRI,-       ;
                        DEFAULT=7,-     ;
                        MIN=0,-         ;
                        MAX=15,-        ;
                        NAME=MINPRPRI,-;
                        SIZE=BYTE,-     ;
                        TYPE=<DYNAMIC,SPECIAL>,-;
                        UNIT=Priority,-
                        VERSION_MASK=<SYSGEN>

;
;  RSRVPAGCNT - granularity, in pages, of page file reservations
;
        PARAMETER       ADDRESS=MMG$GL_RSRVPAGCNT,-     ;
                        DEFAULT=2048,-  ;
                        MIN=128,-               ;
                        MAX=16384,-     ;
                        NAME=RSRVPAGCNT,-;
                        SIZE=LONG,-     ;
                        TYPE=<DYNAMIC,SPECIAL>,-;
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>

;
; WINDOW_SYSTEM SYSGEN parameter for DECwindows
; Specifies default windowing system for a workstation.
; 0 = no windowing system defined, 1=DECW, 2 = VWS
;
        PARAMETER       ADDRESS=EXE$GL_WINDOW_SYSTEM,-  ;
                        DEFAULT=0,-     ; default to no window system
                        NAME=WINDOW_SYSTEM,-;
                        SIZE=LONG,-     ;
                        MIN=0,-         ;
                        MAX=2,-         ; for now only 0, 1 and 2 defined
                        TYPE=<SYS,DYNAMIC>,-;
                        UNIT=Pure-number,-
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 96
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (1)

                        VERSION_MASK=<SYSGEN>

;
; Parameters to controlthe breaking of implicit affinity.
; Controls are on the number of times a process will be skipped before being
; moved and how long it will remain on  the compute queue.
;
        PARAMETER       ADDRESS=SCH$GL_AFFINITY_SKIP,-
                        DEFAULT=2,-
                        MIN=0,-
                        MAX=255,-
                        NAME=AFFINITY_SKIP,-
                        SIZE=LONG,-
                        TYPE=<DYNAMIC,SPECIAL>,-
                        UNIT=Pure-number,-
                        VERSION_MASK=<SYSGEN>

        PARAMETER       ADDRESS=SCH$GL_AFFINITY_TIME,-
                        DEFAULT=0,-
                        MIN=0,-
                        MAX=-1,-
                        NAME=AFFINITY_TIME,-
                        SIZE=LONG,-
                        TYPE=<DYNAMIC,SPECIAL>,-
                        UNIT=10Ms,-
                        VERSION_MASK=<SYSGEN>

;
; Control the number of pages per Error Log Buffer
;
        PARAMETER       ADDRESS=EXE$GB_ERLBUFPAGES,-
                        DEFAULT=2,-
                        MIN=2,-
                        MAX=32,-
                        NAME=ERLBUFFERPAGES,-
                        SIZE=BYTE,-
                        TYPE=<SYS>,-
                        UNIT=Pages,-
                        VERSION_MASK=<SYSGEN>


        .IF     NOT_DEFINED VERSION
        .IIF    NOT_DEFINED GETSYISW,   .ALIGN  LONG
        .ENDC

;
; Define a parameter which determines the tape device allocation class for
; this system.  The tape device allocation class is used to derive a common
; lock resource name for multiple access paths to the same tape device.
;
        PARAMETER       ADDRESS=CLU$GL_TAPE_ALLOCLS, -
                        NAME=TAPE_ALLOCLASS, -
                        TYPE=<CLUSTER>, -
                        DEFAULT=0, -
                        MIN=0, -
                        MAX=255, -
                        UNIT=Pure-number,-
```

```
 5301                          VERSION_MASK=<SYSGEN>
 5302
 5303
 5304 ;*****************************************************************************
 5305 ;*                                                                           *
 5306 ;*           Add new parameters immediately preceding this point.            *
 5307 ;*                                                                           *
 5308 ;*****************************************************************************
 5309
 5310         .IF     NOT_DEFINED GETSYISW
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 98
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (2)

 5312          .SBTTL  COMPUTED VALUES
 5313 ;
 5314 ; The following values are not SYSGEN parameters.  They are included in this
 5315 ; module because it is a convenient way to pass values calculated by SYSBOOT
 5316 ; through to INIT and the running system.
 5317 ;
 5318
 5319          .IF     NOT_DEFINED,VERSION
 5320          .ALIGN  PAGE
 5321
 5322          .IF     NDF,PRMSW               ; For exec version:
 5323 SGN$A_COMPVALUES::                       ; Define base of computed values
 5324          .ENDC
 5325
 5326          .ALIGN  LONG                    ;
 5327          .IF_FALSE
 5328
 5329          DEFINE  SGN$A_COMPVALUES,-
 5330                  VERSION_MASK=<SYSGEN>
 5331
 5332          .ENDC
 5333
 5334          DEFINE  SWP$GL_SHELLSIZ,-       ; PAGES REQUIRED FOR SHELL
 5335                  VERSION_MASK=<SYSGEN>
 5336          .LONG   0                       ;
 5337
 5338          DEFINE  SWP$GW_BAKPTE,-         ; PHD PAGES FOR BAK+WSLX+LCK+VAL
 5339                  VERSION_MASK=<SYSGEN>
 5340          .WORD   0                       ;
 5341
 5342          DEFINE  SWP$GW_EMPTPTE,-        ; EMPTY PHDPAGES
 5343                  VERSION_MASK=<SYSGEN>
 5344          .WORD   0                       ;
 5345
 5346          DEFINE  SWP$GW_WSLPTE,-         ; PHD PAGES FOR FIXED+WSL+PST
 5347                  VERSION_MASK=<SYSGEN>
 5348          .WORD   0                       ;
 5349
 5350          DEFINE  SWP$GB_SHLP1PT,-        ; P1 PAGE TABLES REQUIRED FOR SHELL
 5351                  VERSION_MASK=<SYSGEN>
 5352          .BYTE   0                       ;
 5353
 5354          .BYTE   0                       ; SPARE
 5355
 5356          DEFINE  SWP$GL_BSLOTSZ,-        ; SIZE OF BALANCE SLOT
 5357                  VERSION_MASK=<SYSGEN>
 5358          .LONG   0                       ;
 5359
 5360 ;        DEFINE  SWP$GL_MAP              ; SWAPPER MAP POINTER
 5361 ;        .LONG   0                       ;
 5362
 5363          DEFINE  SWP$GL_PHDBASVA,-       ; BASE ADDRESS OF PHD WINDOW
 5364                  VERSION_MASK=<SYSGEN>
 5365          .LONG   0                       ;
 5366
 5367          DEFINE  SGN$GL_PHDAPCNT,-       ; TOTAL SHELL HEADER PAGES
 5368                  VERSION_MASK=<SYSGEN>
```

```
5369          .LONG   0                       ;
5370
5371          DEFINE  SGN$GL_PHDLWCNT,-       ; COUNT OF LONGWORDS IN PHD
5372                  VERSION_MASK=<SYSGEN>
5373          .LONG   0                       ;
5374
5375          DEFINE  SGN$GL_P1LWCNT,-        ; COUNT OF LW TO END OF P1 PAGETABLE
5376                  VERSION_MASK=<SYSGEN>
5377          .LONG   0                       ;
5378
5379          DEFINE  SGN$GL_PHDPAGCT,-       ; TOTAL PHD PAGES LESS PAGE TABLES
5380                  VERSION_MASK=<SYSGEN>
5381          .LONG   0                       ;
5382
5383          DEFINE  SGN$GL_PTPAGCNT,-       ; TOTAL PAGE TABLE COUNT
5384                  VERSION_MASK=<SYSGEN>
5385          .LONG   0                       ;
5386
5387          DEFINE  MMG$GL_CTLBASVA,-       ; BASE ADDRESS IN CONTROL REGION
5388                  VERSION_MASK=<SYSGEN>
5389          .LONG   0                       ;
5390
5391          DEFINE  EXE$GL_INTSTK,-         ; BASE OF INTERRUPT STACK
5392                  VERSION_MASK=<SYSGEN>
5393          .LONG   0                       ;
5394 ;
5395 ;        THE PRECEDING TWO LONG WORDS MUST BE ADJACENT.
5396 ;
5397
5398          DEFINE  MMG$GL_GPTBASE,-        ; GLOBAL PAGE TABLE BASE ADDRESS
5399                  VERSION_MASK=<SYSGEN>
5400          .LONG   0                       ;
5401
5402          DEFINE  MMG$GL_GPTE,-           ; BASE ADDRESS OF SPT PTES FOR GPT
5403                  VERSION_MASK=<SYSGEN>
5404          .LONG   0                       ; PAGES
5405
5406          DEFINE  MMG$GL_MAXGPTE,-        ; HIGHEST GPTE ADDRESS
5407                  VERSION_MASK=<SYSGEN>
5408          .LONG   0                       ;
5409
5410          DEFINE  MMG$GL_MAXSYSVA,-       ; HIGHEST SYSTEM VA (+1)
5411                  VERSION_MASK=<SYSGEN>
5412          DEFINE  MMG$GL_FRESVA,-         ; SYNONYM
5413                  VERSION_MASK=<SYSGEN>
5414          .LONG   0                       ;
5415
5416          DEFINE  MMG$GL_SPTBASE,-        ; BASE ADDRESS OF SPT (VIRTUAL)
5417                  VERSION_MASK=<SYSGEN>
5418          .LONG   0                       ;
5419
5420          DEFINE  LDR$GL_SPTBASE,-        ; BASE ADDRESS OF SPT - EITHER
5421                  VERSION_MASK=<SYSGEN>
5422          .LONG   0                       ; PHYS OR VIRTUAL AS REQUIRED BY
5423                                          ; SYSLDR
5424          DEFINE  MMG$GL_SPTLEN,-         ; LENGTH OF SYSTEM PAGE TABLE
5425                  VERSION_MASK=<SYSGEN>
```

```
5426          .LONG   0                       ;
5427
5428          DEFINE  MMG$GL_SYSPHD,-         ; VA OF SYSTEM PHD
5429                  VERSION_MASK=<SYSGEN>
5430          .LONG   0                       ;
5431
5432          DEFINE  MMG$GL_SYSPHDLN,-       ; SIZE OF SYSTEM PHD IN BYTES
5433                  VERSION_MASK=<SYSGEN>
5434          .LONG   0                       ;
5435
5436          DEFINE  SWP$GL_BALBASE,-        ; BASE VA OF BALANCE SLOTS FOR
5437                  VERSION_MASK=<SYSGEN>
5438          .LONG   0                       ; PROCESS HEADERS
5439
5440          DEFINE  SWP$GL_BALSPT,-         ; BASE VA IN SPT FOR MAPPING BALANCE
5441                  VERSION_MASK=<SYSGEN>
5442          .LONG   0                       ; SLOTS
5443
5444          DEFINE  MMG$GL_SBR,-            ; SYSTEM BASE REGISTER
5445                  VERSION_MASK=<SYSGEN>
5446          .LONG   0                       ;
5447
5448          DEFINE  MMG$GL_NPAGEDYN,-       ; VA OF NON-PAGED POOL
5449                  VERSION_MASK=<SYSGEN>
5450          .LONG   0                       ;
5451
5452          DEFINE  MMG$GL_NPAGNEXT,-       ; NEXT VA FOR NON-PAGED POOL EXTENSION
5453                  VERSION_MASK=<SYSGEN>
5454          .LONG   0                       ;
5455
5456          DEFINE  MMG$GL_IRPNEXT,-        ; NEXT VA FOR IRP LIST EXTENSION
5457                  VERSION_MASK=<SYSGEN>
5458          .LONG   0                       ;
5459
5460          DEFINE  MMG$GL_LRPNEXT,-        ; NEXT VA FOR LRP LIST EXTENSION
5461                  VERSION_MASK=<SYSGEN>
5462          .LONG   0                       ;
5463
5464          DEFINE  MMG$GL_SRPNEXT,-        ; NEXT VA FOR SRP LIST EXTENSION
5465                  VERSION_MASK=<SYSGEN>
5466          .LONG   0                       ;
5467
5468          DEFINE  MMG$GL_PAGEDYN,-        ; VA OF PAGED POOL
5469                  VERSION_MASK=<SYSGEN>
5470          .LONG   0                       ;
5471
5472          DEFINE  MMG$GL_MAXPFN,-         ; MAXIMUM PFN FOR SYSTEM
5473                  VERSION_MASK=<SYSGEN>
5474          .LONG   0                       ;
5475
5476          DEFINE  MMG$GL_MINPFN,-         ; MINIUMUM PFN IN PFN DATABASE
5477                  VERSION_MASK=<SYSGEN>
5478          .LONG   0                       ;
5479
5480          DEFINE  MMG$GL_MAXMEM,-         ; HIGHEST PFN MAPPED BY SYSBOOT
5481                  VERSION_MASK=<SYSGEN>
5482          .LONG   0                       ; INCLUDES PAGES NOT IN PFN DATABASE
```

```
5483
5484        DEFINE  EXE$GL_RPB,-            ; VIRTUAL ADDRESS OF RESTART PARAMETER BLK
5485                VERSION_MASK=<SYSGEN>
5486        .LONG   0                       ;
5487
5488        DEFINE  EXE$GL_SCB,-            ; VIRTUAL ADDRESS OF SCB
5489                VERSION_MASK=<SYSGEN>
5490        .LONG   0                       ;
5491
5492        DEFINE  EXE$GL_ARCHFLAG,-       ; ARCHITECTURAL FLAGS (BITS DEFINED
5493                VERSION_MASK=<SYSGEN>
5494        .LONG   0                       ; BY $ARCDEF
5495
5496        DEFINE  EXE$GL_STATE,-          ; FLAGS BOOTSTRAP PROGRESS
5497                VERSION_MASK=<SYSGEN>
5498        .LONG   0
5499
5500        DEFINE  LDR$GL_FREE_PT,-        ; POINTER TO FREE SPTE
5501                VERSION_MASK=<SYSGEN>
5502        .LONG   0,0
5503
5504        DEFINE  EXE$GB_CPUDATA,-        ; 16 BYTES OF DATA ABOUT THE CPU
5505                VERSION_MASK=<SYSGEN>
5506        .LONG   0,0,0,0                 ; 4 BYTES FOR SID, REST CPU SPECIFIC
5507
5508        DEFINE  EXE$GB_CPUTYPE,-        ; CPU TYPE READ FROM SID
5509                VERSION_MASK=<SYSGEN>
5510        .BYTE   0                       ;
5511
5512        DEFINE  EXE$GW_CPUMODEL,-       ; CPU MODEL NUMBER
5513                VERSION_MASK=<SYSGEN>
5514        .WORD   0
5515
5516        .IF     NOT_DEFINED VERSION
5517        .ALIGN  LONG
5518        .IF_FALSE
5519
5520        DEFINE  CLU$GL_NISCS_AUTH,-
5521                VERSION_MASK=<SYSGEN>
5522
5523        DEFINE  PFN$C_WORD_LEN,-
5524                VERSION_MASK=<SYSGEN>
5525
5526        DEFINE  PFN$C_LONG_LEN,-
5527                VERSION_MASK=<SYSGEN>
5528
5529        .ENDC
5530
5531        DEFINE  CLU$GB_NISCS_COMM,-     ; NISCS communications region
5532                VERSION_MASK=<SYSGEN>
5533        DEFINE  CLU$GQ_NISCS_AUTH,-     ;  Authorization quadword
5534                VERSION_MASK=<SYSGEN>
5535        .QUAD   0
5536        DEFINE  CLU$GL_NISCS_GROUP,-    ;  Group code
5537                VERSION_MASK=<SYSGEN>
5538        .LONG   0
5539        .LONG   0                       ;  Spare for NISCS extensions
```

```
5540          .QUAD   0                       ;  Spare for NISCS extensions
5541
5542          .IF     NOT_DEFINED,VERSION
5543          .IF     NDF,PRMSW               ; Size of communcations region
5544          CLU$GS_NISCS_COMM == 24
5545          PFN$C_WORD_LEN == 18            ;  ... with word length FLINK and BLINK
5546          PFN$C_LONG_LEN == 22            ;  ... with longword length FLINK and BLINK
5547          .ENDC                           ;
5548          .IF_FALSE
5549
5550          DEFINE  CLU$GS_NISCS_COMM,-
5551                  VERSION_MASK=<SYSGEN>
5552
5553          DEFINE  PFN$C_WORD_LEN,-
5554                  VERSION_MASK=<SYSGEN>
5555
5556          DEFINE  PFN$C_LONG_LEN,-
5557                  VERSION_MASK=<SYSGEN>
5558
5559          .ENDC
5560
5561          DEFINE  PFN$GB_LENGTH,-         ; Number of bytes per page in PFN data base
5562                  VERSION_MASK=<SYSGEN>
5563          .BYTE   PFN$C_WORD_LEN          ; Defaults to word length FLINK and BLINK
5564
5565          DEFINE  MMG$GW_BIGPFN,-                 ; Flag to indicate size of PFN FLINK
5566                  VERSION_MASK=<SYSGEN>
5567          .WORD   0                       ; word for historical reasons
5568
5569          DEFINE  EXE$GW_PGFL_FID,-       ; FILE ID OF PAGEFILE.SYS
5570                  VERSION_MASK=<SYSGEN>
5571          .WORD   0,0,0                   ; IF FILE IS IN PAGE FILE
5572
5573          .IF     NOT_DEFINED,VERSION
5574          .ALIGN  LONG                    ; LONWORD ALIGN POINTERS
5575          .ENDC
5576          DEFINE  PFN$A_BASE,-            ; BASE OF PFN POINTERS
5577                  VERSION_MASK=<SYSGEN>
5578          PFNALC  L,PTE,-                 ; ADDRESS OF PAGE TABLE ENTRY
5579                  VERSION_MASK=<SYSGEN>
5580          PFNALC  L,BAK,-                 ; BACKING STORE ADDRESS
5581                  VERSION_MASK=<SYSGEN>
5582          PFNALC  W,REFCNT,-              ; REFERENCE COUNT
5583                  VERSION_MASK=<SYSGEN>
5584          PFNALC  x,<FLINK,-              ; FORWARD LINK
5585                  SHRCNT>,-                       ; ALSO USED AS GLOBAL SHARE COUNT
5586                  VERSION_MASK=<SYSGEN>
5587          PFNALC  x,<BLINK,-              ; BACK LINK
5588                  WSLX>,-                 ; ALSO USED AS WORKING SET LIST INDEX
5589                  VERSION_MASK=<SYSGEN>
5590          PFNALC  W,SWPVBN,-              ; SWAP IMAGE VIRTUAL BLOCK OFFSET
5591                  VERSION_MASK=<SYSGEN>
5592          PFNALC  B,STATE,-               ; STATE OF PAGE
5593                  VERSION_MASK=<SYSGEN>
5594          PFNALC  B,TYPE,-                ; TYPE OF PAGE
5595                  VERSION_MASK=<SYSGEN>
5596
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 103
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (2)

 5597          DEFINE  EXE$GT_STARTUP,-        ; NAME OF STARTUP COMMAND FILE
 5598                  VERSION_MASK=<SYSGEN>
 5599          .ASCIC  /SYS$SYSTEM:STARTUP.COM/; DEFAULT VALUE
 5600          .IF     NOT_DEFINED VERSION
 5601          .BLKB   <32-<.-EXE$GT_STARTUP>> ; ALLOW FOR 31 BYTES + COUNT
 5602          .ENDC
 5603
 5604
 5605
 5606          .IF     NOT_DEFINED VERSION
 5607          .IF     NDF,PRMSW               ; IF EXEC VERSION
 5608
 5609 ;
 5610 ;  The cells in this module between the definition of EXE$C_SYSPARSZ and
 5611 ;  the definition of BOO$C_SYSPARSZ are used for communication between
 5612 ;  SYSBOOT and SYS.EXE without interference from the SYSGEN USE and WRITE
 5613 ;  commands. SYSBOOT uses BOO$C_SYSPARSZ as the size of the parameter area.
 5614 ;  SYSGEN uses EXE$C_SYSPARSZ as its size constant and so SYSGEN commands
 5615 ;  do not affect the contents of cells that follow the definition of
 5616 ;  EXE$C_SYSPARSZ.
 5617 ;
 5618
 5619 EXE$C_SYSPARSZ==.-EXE$A_SYSPARAM        ; SIZE OF SYSTEM PARAMETERS
 5620 BOO$C_SYSPARSZ==.-EXE$A_SYSPARAM        ; SIZE OF PARAMETER AREA READ BY SYSBOOT
 5621 ;
 5622 ; Define offsets to selected data cells from the base of the non-SYSGEN
 5623 ; data defined in this module.
 5624 ;
 5625 PFN_AB_STATE == PFN$AB_STATE - SGN$A_COMPVALUES
 5626 PFN_AB_TYPE == PFN$AB_TYPE - SGN$A_COMPVALUES
 5627 PFN_AL_BAK == PFN$AL_BAK - SGN$A_COMPVALUES
 5628 PFN_AL_PTE == PFN$AL_PTE - SGN$A_COMPVALUES
 5629 PFN_AW_REFCNT == PFN$AW_REFCNT - SGN$A_COMPVALUES
 5630 PFN_AW_SWPVBN == PFN$AW_SWPVBN - SGN$A_COMPVALUES
 5631 PFN_AX_BLINK == PFN$AX_BLINK - SGN$A_COMPVALUES
 5632 PFN_AX_FLINK == PFN$AX_FLINK - SGN$A_COMPVALUES
 5633 PFN_AX_SHRCNT == PFN$AX_SHRCNT - SGN$A_COMPVALUES
 5634 PFN_AX_WSLX == PFN$AX_WSLX - SGN$A_COMPVALUES
 5635 MMG_GL_GPTBASE == MMG$GL_GPTBASE - SGN$A_COMPVALUES
 5636 MMG_GL_SPTBASE == MMG$GL_SPTBASE - SGN$A_COMPVALUES
 5637 SWP_GL_BALSPT == SWP$GL_BALSPT - SGN$A_COMPVALUES
 5638 SWP_GL_BALBASE == SWP$GL_BALBASE - SGN$A_COMPVALUES
 5639
 5640          .IFF                            ;
 5641          .PSECT  $$$918,LONG             ;
 5642          .LONG   0                       ; FLAG TO MARK END
 5643 BOO$C_PRMBLKSZ==.-BOO$A_PRMBLK          ; Compute length of descriptor area
 5644          .ENDC                           ;
 5645          .IF_FALSE
 5646
 5647          DEFINE  EXE$C_SYSPARSZ,-
 5648                  VERSION_MASK=<SYSGEN>
 5649
 5650          DEFINE  BOO$C_SYSPARSIZ,-
 5651                  VERSION_MASK=<SYSGEN>
 5652
 5653          DEFINE  PFN_AB_STATE,-
```

```
5654                    VERSION_MASK=<SYSGEN>
5655
5656            DEFINE  PFN_AB_TYPE,-
5657                    VERSION_MASK=<SYSGEN>
5658
5659            DEFINE  PFN_AL_BAK,-
5660                    VERSION_MASK=<SYSGEN>
5661
5662            DEFINE  PFN_AL_PTE,-
5663                    VERSION_MASK=<SYSGEN>
5664
5665            DEFINE  PFN_AW_REFCNT,-
5666                    VERSION_MASK=<SYSGEN>
5667
5668            DEFINE  PFN_AW_SWPVBN,-
5669                    VERSION_MASK=<SYSGEN>
5670
5671            DEFINE  PFN_AX_BLINK,-
5672                    VERSION_MASK=<SYSGEN>
5673
5674            DEFINE  PFN_AX_FLINK,-
5675                    VERSION_MASK=<SYSGEN>
5676
5677            DEFINE  PFN_AX_SHRCNT,-
5678                    VERSION_MASK=<SYSGEN>
5679
5680            DEFINE  PFN_AX_WSLX,-
5681                    VERSION_MASK=<SYSGEN>
5682
5683            DEFINE  MMG_GL_GPTBASE,-
5684                    VERSION_MASK=<SYSGEN>
5685
5686            DEFINE  MMG_GL_SPTBASE,-
5687                    VERSION_MASK=<SYSGEN>
5688
5689            DEFINE  SWP_GL_BALSPT,-
5690                    VERSION_MASK=<SYSGEN>
5691
5692            DEFINE  SWP_GL_BALBASE,-
5693                    VERSION_MASK=<SYSGEN>
5694
5695            DEFINE  BOO$C_PRMBLKSZ,-
5696                    VERSION_MASK=<SYSGEN>
5697
5698            .ENDC   ; Version
5699            .ENDC           ; NOT_DEFINED GETSYISW
5700
5701 ;
5702 ; Terminate the definition of the macro
5703 ;
5704
5705            .ENDM   SYI_GENERATE_TABLE
5706
5707 ;
5708 ; Invoke the macro just defined (if this isn't getsyi)
5709 ;
5710
```

```
SYSPARAM - SYSTEM PARAMETERS 10-MAY-1989 15:26:20 VAX MACRO V5.0-8 Page 105
X-101U18 SYSGEN PARAMETERS 6-APR-1989 10:18:35 [SYS.SRC]SYSPARAM.MAR;2 (2)

 5711          .IIF     NOT_DEFINED GETSYISW,    SYI_GENERATE_TABLE
```

```
 5712
 5713          .IIF    NOT_DEFINED GETSYISW,    .END     ; PREFIX FILE, IF GETSYISW
```

9 SYSTEM_SERVICE_DISPATCHER.LIS

```
    SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 0
    Table of contents

      (2)       102         Declarations
      (3)       165         CMOD$SSVECX - RMS exception vector
      (4)       199         - Filtered EXEC Mode Dispatcher
      (5)       247         - Change Mode to EXEC Dispatcher
      (6)       384         Error Handling for EXEC Mode Services
      (7)       412         CMOD$ASTEXIT - System Service to Clear AST Context
      (8)       466         - Filtered KERNEL Mode Dispatcher
      (9)       514         - Change Mode to KERNEL Dispatcher
      (10)      659         Error Handling for KERNEL Mode Services
      (11)      687         SERVICE_EXIT - System Service Completion
      (12)      759         INHEXCP - Inhibited CHMK or CHME code handling
```

```
SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 1
X-12 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER.MAR;1 (1)

     1          .TITLE  SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher
     2          .IDENT  'X-12'
     3
     4  ;******************************************************************************
     5  ;*                                                                            *
     6  ;*  COPYRIGHT (c) 1985 BY                                                     *
     7  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
     8  ;*  ALL RIGHTS RESERVED.                                                      *
     9  ;*                                                                            *
    10  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
    11  ;*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE            *
    12  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER       *
    13  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
    14  ;*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY         *
    15  ;*  TRANSFERRED.                                                              *
    16  ;*                                                                            *
    17  ;*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE      *
    18  ;*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT          *
    19  ;*  CORPORATION.                                                              *
    20  ;*                                                                            *
    21  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS       *
    22  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
    23  ;*                                                                            *
    24  ;*                                                                            *
    25  ;******************************************************************************
    26
    27  ;++
    28  ; Facility:
    29  ;
    30  ;       VAX/VMS Executive, EXCEPTION image
    31  ;
    32  ;       System Service Dispatcher for Kernel and Exec Access Modes
    33  ;
    34  ; Abstract:
    35  ;
    36  ;       The code in this module contains the system service dispatchers for
    37  ;       exec and kernel access modes. In addition, this module contains the
    38  ;       kernel mode dispatcher that is used in attached processors in an
    39  ;       asymmetric multiprocessor.
    40  ;
    41  ; Author:
    42  ;
    43  ;       David N. Cutler
    44  ;
    45  ; Creation Date:
    46  ;
    47  ;       22 June 1976
    48  ;
    49  ; Modified by:
    50  ;
    51  ;       X-12    HH0322          Hai Huang               23-Jun-1988
    52  ;               Make SERVICE_EXIT a global symbol so the condition handler
    53  ;               search algorithm can detect a system service frame.
    54  ;
    55  ;       X-11    RNG5011         Rod Gamache             16-Apr-1987
    56  ;               Speed up certain operations. Like using CTL$GL_PCB and
    57  ;               interleaving register and memory references in CMOD$ASTEXIT.
```

```
;
;       X-10    SF04006         Stephen Fiorelli        24-Feb-1987
;               Make the dispatching path go a little faster.
;               Intersperse memory and register references for speed,
;               and eliminated special case code incase we are the
;               swapper.  The swapper now has a p1 space.
;
;       X-9     SF04005         Stephen Fiorelli        26-Sep-1986
;               Do not continue on in the error path of SERVICE_EXIT
;               if the stack pointer is in system space.  This implies
;               the swapper which does not have a P1 space, and cannot
;               reference P1 cells.
;
;       X-8     SF04004         Stephen Fiorelli        09-Apr-1986
;               Add a special case CHME code #0, used to generate
;               a possible system service failure exception that
;               occurred during the execution of rms synchronization
;               code.
;
;       X-7     SF04003         Stephen Fiorelli        21-Mar-1986
;               Load PCB before determining if a CHMK code is for
;               a user written system service.
;
;       X-6     SF04002         Stephen Fiorelli        17-Dec-1985
;               Add some code that will change the return PC pushed
;               onto the stack before a system service is executed.
;               This is done because some services need to perform
;               some synchronization, before returning.  The changed
;               PC will represent the address of the synchronization
;               routine the system service needs to execute.
;
;       X-4     TCM0001         Trudy C. Matthews       5-Nov-1985
;               Change default alignment of psect EXEC$NONPAGED_CODE to QUAD.
;
;       V04-002 SF04001         Stephen Fiorelli        16-Oct-1985
;               Change PSECT declaration to be a standard PSECT.
;
;       V04-001 LJK4007         Lawrence J. Kenah       15-May-1985
;               Being a complete rewrite of this part of the system.
;               The module was split into two functional pieces. One piece
;               contains the definitions of the various services. The other
;               piece, this module, contains the various dispatchers.
;
```

```
SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 3
X-12 Declarations 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER.MAR;1 (2)

                .SUBTITLE       Declarations

; Include files:

                $DISPDEF                        ; Dispatch vector contents
                $IPLDEF                         ; IPL symbolic definitions
                $PCBDEF                         ; Process control block offsets
                $PRDEF                          ; Processor register definitions
                $PSLDEF                         ; Fields in processor status longword


; External declarations

                .DISABLE        GLOBAL

; Status codes

                .EXTERNAL       -
                        BUG$_MTXCNTNONZ, -
                        SS$_ACCVIO, -
                        SS$_ILLSER, -
                        SS$_INHCHME, -
                        SS$_INHCHMK, -
                        SS$_INSFARG

; Global data cells

                .EXTERNAL       -               ; ... for exec mode services
                        CMOD$GW_CHME_LIMIT, -
                        CMOD$AR_EXEC_DISPATCH_VECTOR, -
                        CMOD$AB_EXEC_INHIBIT_MASK, -
                        CMOD$AB_EXEC_MIN_ARG_COUNT

                .EXTERNAL       -               ; ... and for kernel mode services
                        CMOD$GW_CHMK_LIMIT, -
                        CMOD$AR_KERNEL_DISPATCH_VECTOR, -
                        CMOD$AB_KERNEL_INHIBIT_MASK, -
                        CMOD$AB_KERNEL_MIN_ARG_COUNT

                .EXTERNAL       -               ; Other global data cells
                        CMOD$AL_EXIT_TYPE ,-
                        CTL$GB_SSFILTER, -
                        CTL$GL_PCB, -
                        CTL$GL_RMSBASE, -
                        CTL$GL_USRCHME, -
                        CTL$GL_USRCHMK, -
                        EXE$GL_USRCHME, -
                        EXE$GL_USRCHMK,-
                        SMP$GL_BASE_MSK

; External routines

                .EXTERNAL       -
                        EXE$REFLECT, -
                        EXE$SSFAIL, -
                        SCH$NEWLVL
```

```
    SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 4
    X-12 Declarations 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER.MAR;1 (2)

      159 ; PSECT Declarations:
      160
      161          .DEFAULT        DISPLACEMENT , WORD
      162
      163          DECLARE_PSECT   EXEC$NONPAGED_CODE,ALIGNMENT=QUAD
```

```
SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 5
X-12 CMOD$SSVECX - RMS exception vector 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER.MAR;1

  165          .SUBTITLE       CMOD$SSVECX - RMS exception vector
  166 ;+
  167 ; Functional Description:
  168 ;
  169 ;       This routine is call to generate a possible system service
  170 ;       exception during the execution of rms synchronization code
  171 ;       (found in the module system_service_exit.mar).  The status
  172 ;       was place in R2, this routine merely places the status in
  173 ;       R0 and returns through the common system service exit
  174 ;       path.
  175 ;
  176 ;       A change mode to exec is needed for this because the
  177 ;       rms synchronization code executes in the callers mode.
  178 ;
  179 ; Input Parameters:
  180 ;
  181 ;       R2 - Status placed there within common rms synchronization
  182 ;            code
  183 ;
  184 ; Output Parameters:
  185 ;
  186 ;       R0 - Contains the status from the rms synch code
  187 ;
  188 ; Side Effects:
  189 ;
  190 ;       none
  191 ;-
  192
  193 CMOD$SSVECX::
  194          JSB     B^10$                   ; Push a PC onto the stack
  195 10$:     MOVAB   SERVICE_EXIT,(SP)       ; Change return PC to common exit path
  196          MOVL    R2,R0                   ; Move status into R0
  197          RSB                             ; Return through service exit path
```

```
    SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 6
    X-12 - Filtered EXEC Mode Dispatcher 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER.MAR;1 (4

      199          .SUBTITLE - Filtered EXEC Mode Dispatcher
      200 ;+
      201 ; Functional Description:
      202 ;
      203 ;       This routine receives control via the system control block (SCB) when
      204 ;       a CHME instruction is executed on a system that has the SSINHIBIT
      205 ;       SYSGEN parameter set. This routine checks whether the calling process
      206 ;       can execute the service before letting control drop through to the
      207 ;       normal CHME dispatcher .
      208 ;
      209 ; Input Parameters:
      210 ;
      211 ;       00(SP) - CHME code
      212 ;       04(SP) - PC following CHME instruction
      213 ;       08(SP) - PSL at the time that CHME was executed.
      214 ;
      215 ; Implicit Input:
      216 ;
      217 ;       CTL$GB_SSFILTER - System service filter for the calling process
      218 ;
      219 ;       Inhibit mask associated with the specific exec mode service
      220 ;
      221 ; Output Parameters:
      222 ;
      223 ;       Unchanged from input
      224 ;
      225 ; Return Status:
      226 ;
      227 ;       If the service is allowed, this routine passes control to EXE$CMODEXEC
      228 ;
      229 ;       Failure Codes
      230 ;               SS$_INHCHME - The calling process is not allowed to
      231 ;                       call this service
      232 ;-
      233
      234          .ALIGN  QUAD                    ; SCB vectors must be aligned
      235
      236 EXE$CMODEXECX::
      237          BICL3   8(SP), -
      238                  #PSL$M_CURMOD, R0       ; Check the previous mode
      239          BNEQ    EXE$CMODEXEC            ; Only USER mode is inhibited
      240          MOVZBL  (SP), R0                ; Pick up the CHME code (MOD 256)
      241          BITB    CMOD$AB_EXEC_INHIBIT_MASK[R0], -
      242                  G^CTL$GB_SSFILTER       ; AND with the inhibit mask
      243          BEQL    EXE$CMODEXEC            ; This code is allowed
      244          MOVZWL  #SS$_INHCHME, R1        ; Set the exception code
      245          BRW     INHEXCP                 ; ... and reflect it
```

```
247          .SUBTITLE - Change Mode to EXEC Dispatcher
248 ;+
249 ; Functional Description:
250 ;
251 ;       This routine receives control via the system control block (SCB) when
252 ;       a CHME instruction is executed. If the CHME code corresponds to an
253 ;       existing service, this routine passes control to a service-specific
254 ;       procedure after building a naked call frame (no saved registers or
255 ;       stack alignment) on the EXEC stack, checking for the required argument
256 ;       count, and probing the argument list for write access.
257 ;
258 ; Input Parameters:
259 ;
260 ;       00(SP) - CHME code
261 ;       04(SP) - PC following CHME instruction
262 ;       08(SP) - PSL at the time that CHME was executed.
263 ;
264 ; Implicit Input:
265 ;
266 ;       Argument list to be passed to the service
267 ;
268 ; Output Parameters:
269 ;
270 ;       none
271 ;
272 ; Implicit Output:
273 ;
274 ;       A call frame is built on the exec stack (by hand for speed)
275 ;
276 ;               00(FP) - Null condition handler address
277 ;               04(FP) - Null register save mask
278 ;               08(FP) - Saved AP
279 ;               12(FP) - Saved FP
280 ;               16(FP) - Return PC. This cell always contains the address
281 ;                        of SERVICE_EXIT.
282 ;
283 ; Return Status:
284 ;
285 ;       Successful passage through this routine result in a control transfer
286 ;       to a service-specific procedure. This routine does not return any
287 ;       success status.
288 ;
289 ;       Failure Codes
290 ;
291 ;               SS$_ACCVIO - The argument list could not be read
292 ;               SS$_INSFARG - Not enough arguments were supplied to the service
293 ;               SS$_ILLSER - An unknown change mode code was used
294 ;
295 ; Note:
296 ;
297 ;       This routine is coded for speed. Instructions that write to memory (to
298 ;       build the call frame) are interspersed with instructions that
299 ;       reference only registers or the instruction stream.
300 ;-
301
302          .ENABLE         LOCAL_BLOCK
303
```

```
    SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 8
    X-12 - Change Mode to EXEC Dispatcher 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER.MAR;1 (

      304          .ALIGN  QUAD                    ; SCB vectors must be aligned
      305
      306 EXE$CMODEXEC::
      307          POPL    R0                      ; Retrieve CHME code from stack
      308          BEQL    CMOD$SSVECX             ; Special code for CHME #0
      309          PUSHAB  SERVICE_EXIT            ; RETURN PC
      310          MOVZBL  R0, R1                  ; Limit the range of CHME codes
      311
      312 ; Build the rest of the call frame.
      313
      314          PUSHL   FP                      ; Save FP
      315          PUSHL   AP                      ; Save AP
      316          MOVAQ   CMOD$AR_EXEC_DISPATCH_VECTOR[R1], -
      317                  R1                      ; Get service-specific data
      318          CLRQ    -(SP)                   ; Fill in the rest of the call frame
      319          MOVL    SP, FP                  ; Point FP at new call frame
      320
      321 ; See if the CHME code corresponds to a service that has been loaded
      322
      323          CMPW    R0, CMOD$GW_CHME_LIMIT
      324          BGEQU   10$
      325
      326 ; Insure that the service has been called with enough arguments and that
      327 ; the argument list is readable from caller's mode.
      328
      329          IFNORD  DISP_W_ARG_LIST_SIZE(R1), -
      330                  (AP), EXEC_ACCVIO       ; Branch if inaccessible argument list
      331
      332          CMPB    (AP), -                 ; Check for enough arguments
      333                  DISP_B_ARGUMENT_COUNT(R1)
      334          BLSSU   EXEC_INSFARG
      335          TSTB    DISP_B_EXIT_TYPE(R1)    ; Check if special synchonization code
      336                                          ; needs to be executed after service calll
      337          BEQL    5$                      ; No special synchronization code
      338
      339 ;
      340 ; Some system services need to execute some form
      341 ; of synchronization code before returning back to
      342 ; the caller.  The following code changes the
      343 ; return PC placed on the stack by CHME
      344 ; to be the address of the synchronization code
      345 ; to be executed.
      346 ;
      347 ; CMOD$AL_EXIT_TYPE - Table containing the addresses of
      348 ;                     synchronization code indexed by
      349 ;                     the contants of DISP_B_EXIT_TYPE(R1)
      350 ;
      351
      352          MOVZBL  DISP_B_EXIT_TYPE(R1),R2         ; Get the index into
      353                                                  ; CMOD$AL_EXIT_TYPE
      354          MOVL    CMOD$AL_EXIT_TYPE[R2],20(SP)    ; Change the return address
      355
      356 5$:      JMP     @DISP_A_SERVICE_ROUTINE(R1)
      357
      358 ; A CHME instruction was executed with a code that does not represent an
      359 ; existing service. Try the various loadable service dispatchers before
      360 ; returning an error status.
```

```
361 ;
362 ; Note that these various dispatchers can either ACCEPT the service code
363 ; (and eventually return with a RET to SERVICE_EXIT) or reject the code. In
364 ; this latter case, they return here with an RSB to allow the next dispatcher
365 ; in line to be tried.
366
367 10$:    TSTB    G^CTL$GB_SSFILTER
368         BNEQ    40$
369
370         MOVL    G^CTL$GL_USRCHME, R1    ;
371         BEQL    20$
372         JSB     (R1)
373
374 20$:    MOVL    G^EXE$GL_USRCHME, R1    ;
375         BEQL    30$
376         JSB     (R1)
377
378 30$:    MOVZWL  #SS$_ILLSER, R0
379         RET
380
381 40$:    MOVZWL  #SS$_INHCHME, R1        ; Load failure code
382         BRW     INHEXCP1                ; Reflect the error
```

```
    SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 10
    X-12 Error Handling for EXEC Mode Services 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER.MP

      384          .SUBTITLE       Error Handling for EXEC Mode Services
      385 ;+
      386 ; Functional Description:
      387 ;
      388 ;       These routine receive control when the change mode dispatcher delects
      389 ;       an error while checking the validity of the argument list.
      390 ;
      391 ; Input Parameters:
      392 ;
      393 ;       Identical to the output of the change mode dispatcher
      394 ;
      395 ; Implicit Output:
      396 ;
      397 ;       If the change mode code is recognized, then the service is immediately
      398 ;       dismissed with an appropriate status. Note that errors detected by the
      399 ;       dispatcher are not signalled as system service failure exceptions.
      400 ;-
      401
      402 EXEC_ACCVIO:
      403          MOVZWL  #SS$_ACCVIO, R0
      404          RET
      405
      406 EXEC_INSFARG:
      407          MOVZWL  #SS$_INSFARG, R0
      408          RET
      409
      410          .DISABLE        LOCAL_BLOCK
```

```
SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 11
X-12 CMOD$ASTEXIT - System Service to Clear A 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER

                .SUBTITLE       CMOD$ASTEXIT - System Service to Clear AST Context
;+
; Functional Description:
;
;       This routine is called after an AST has completed to clear the AST
;       active bit and to allow any pending ASTs to be delivered. A system
;       service is required to accomplish this because an AST executes in
;       caller's mode but the AST context is stored in the PCB.
;
;       This service can be called directly with a naked CHMK #ASTEXIT or
;       through the EXE$CLRAST system service vector. The AST delivery code
;       uses the forst method to avoid cluttering the stack with an additional
;       call frame during AST exit processing.
;
; Input Parameters:
;
;       0(SP) - PC following the CHMK #ASTEXIT
;       4(SP) - PSL at the time that the CHMK #ASTEXIT was issued
;
; Output Parameters:
;
;       none
;
; Implicit Output:
;
;       PCB$B_ASTACT is cleared for caller's mode.
;
;       PHD$B_ASTLVL is set to the access mode of the next pending AST, if
;       there is one, or to 4 if there are no pending ASTs that can be
;       delivered at the current time.
;
; Side Effects:
;
;       The calling thread is demoted from execution at AST level to execution
;       at main line level. In other words, this service can be used by code
;       executing at AST level to allow other ASTs to be delivered before the
;       caller exits its own thread of execution.
;-

CMOD$ASTEXIT::
        EXTZV   #PSL$V_CURMOD, -        ; Load caller's mode into R0
                #PSL$S_CURMOD, -
                4(SP), R0
        PUSHL   R4                      ; Save R4
        MOVL    G^CTL$GL_PCB,R4         ; Get PCB address of caller
        PUSHL   R2                      ; ... and R2
        SETIPL  #IPL$_ASTDEL            ; Prevent AST delivery
        BBCCI   R0, PCB$B_ASTACT(R4), -
                10$                     ; Clear AST active bit for caller's mode
10$:    JSB     G^SCH$NEWLVL            ; Compute new AST level setting
        POPL    R2                      ; Restore R2
        POPL    R4                      ; ... and R4
        REI                             ; and return to caller
```

```
    SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 12
    X-12 - Filtered KERNEL Mode Dispatcher 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER.MAR;1

      466          .SUBTITLE - Filtered KERNEL Mode Dispatcher
      467 ;+
      468 ; Functional Description:
      469 ;
      470 ;       This routine receives control via the system control block (SCB) when
      471 ;       a CHMK instruction is executed on a system that has the SSINHIBIT
      472 ;       SYSGEN parameter set. This routine checks whether the calling process
      473 ;       can execute the service before letting control drop through to the
      474 ;       normal CHMK dispatcher .
      475 ;
      476 ; Input Parameters:
      477 ;
      478 ;       00(SP) - CHMK code
      479 ;       04(SP) - PC following CHMK instruction
      480 ;       08(SP) - PSL at the time that CHMK was executed.
      481 ;
      482 ; Implicit Input:
      483 ;
      484 ;       CTL$GB_SSFILTER - System service filter for the calling process
      485 ;
      486 ;       Inhibit mask associated with the specific exec mode service
      487 ;
      488 ; Output Parameters:
      489 ;
      490 ;       Unchanged from input
      491 ;
      492 ; Return Status:
      493 ;
      494 ;       If the service is allowed, this routine passes control to EXE$CMODKRNL
      495 ;
      496 ;       Failure Codes
      497 ;               SS$_INHCHMK - The calling process is not allowed to
      498 ;                       call this service
      499 ;-
      500
      501          .ALIGN  QUAD                    ; SCB vectors must be aligned
      502
      503 EXE$CMODKRNLX::
      504          BICL3   8(SP), -
      505                  #PSL$M_CURMOD, R0       ; Check the previous mode
      506          BNEQ    EXE$CMODKRNL            ; Only USER mode is inhibited
      507          MOVZBL  (SP), R0                ; Pick up the CHMK code
      508          BITB    CMOD$AB_KERNEL_INHIBIT_MASK[R0], -
      509                  G^CTL$GB_SSFILTER       ; AND with inhibit mask
      510          BEQL    EXE$CMODKRNL            ; This code is allowed
      511          MOVZWL  #SS$_INHCHMK, R1        ; Set the exception code
      512          BRW     INHEXCP                 ; ... and reflect it
```

```
514          .SUBTITLE - Change Mode to KERNEL Dispatcher
515 ;+
516 ; Functional Description:
517 ;
518 ;       This routine receives control via the system control block (SCB) when
519 ;       a CHMK instruction is executed. If the CHMK code corresponds to an
520 ;       existing service, this routine passes control to a service-specific
521 ;       procedure after building a naked call frame (no saved registers or
522 ;       stack alignment) on the EXEC stack, checking for the required argument
523 ;       count, and probing the argument list for write access.
524 ;
525 ; Input Parameters:
526 ;
527 ;       00(SP) - CHMK code
528 ;       04(SP) - PC following CHMK instruction
529 ;       08(SP) - PSL at the time that CHMK was executed.
530 ;
531 ; Implicit Input:
532 ;
533 ;       Argument list to be passed to the service
534 ;
535 ; Output Parameters:
536 ;
537 ;       none
538 ;
539 ; Implicit Output:
540 ;
541 ;       A call frame is built on the kernel stack (by hand for speed)
542 ;
543 ;               00(FP) - Null condition handler address
544 ;               04(FP) - Null register save mask
545 ;               08(FP) - Saved AP
546 ;               12(FP) - Saved FP
547 ;               16(FP) - Return PC. This cell always contains the address
548 ;                        of SERVICE_EXIT.
549 ;
550 ; Return Status:
551 ;
552 ;       Successful passage through this routine result in a control transfer
553 ;       to a service-specific procedure. This routine does not return any
554 ;       success status.
555 ;
556 ;       Failure Codes
557 ;
558 ;               SS$_ACCVIO - The argument list could not be read
559 ;               SS$_INSFARG - Not enough arguments were supplied to the service
560 ;               SS$_ILLSER - An unknown change mode code was used
561 ;
562 ; Note:
563 ;
564 ;       This routine is coded for speed. Instructions that write to memory (to
565 ;       build the call frame) are interspersed with instructions that
566 ;       reference only registers or the instruction stream.
567 ;-
568
569          .ENABLE         LOCAL_BLOCK
570
```

```
    SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 14
    X-12 - Change Mode to KERNEL Dispatcher 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER.MAR;1

                    .ALIGN  QUAD                    ; SCB vectors must be aligned
                                                    ;
EXE$CMODKRNL::                                      ;
                    POPL    R0                      ; Retrieve CHMK code from stack
                    BEQL    CMOD$ASTEXIT            ; Special code for CHMK #0
                    PUSHAB  SERVICE_EXIT            ; RETURN PC
                    MOVZBL  R0, R1                  ; Limit the range of CHMK codes
                                                    ;
; Build the rest of the call frame.                 ;
                                                    ;
                    MOVL    G^CTL$GL_PCB,R4         ; Get current PCB address
                    PUSHL   FP                      ; Save FP
                    PUSHL   AP                      ; Save AP
                    MOVAQ   CMOD$AR_KERNEL_DISPATCH_VECTOR[R1], -
                            R1                      ; Get service-specific data
                    CLRQ    -(SP)                   ; Fill in the rest of the call frame
                    MOVL    SP, FP                  ; Point FP at new call frame
                                                    ;
                                                    ;
; See if the CHMK code corresponds to a service that has been loaded
                                                    ;
                    CMPW    R0, CMOD$GW_CHMK_LIMIT
                    BGEQU   10$
                                                    ;
; Insure that the service has been called with enough arguments and that
; the argument list is readable from caller's mode.
                                                    ;
                                                    ;
                    IFNORD  DISP_W_ARG_LIST_SIZE(R1), -
                            (AP), KERNEL_ACCVIO     ; Branch if inaccessible argument list
                                                    ;
                    CMPB    (AP), -                 ; Check for enough arguments
                            DISP_B_ARGUMENT_COUNT(R1)
                    BLSSU   KERNEL_INSFARG
                    TSTB    DISP_B_EXIT_TYPE(R1)    ; Check if special synchonization code
                                                    ; needs to be executed after service calll
                    BEQL    5$                      ; No special synchronization code
                                                    ;
;
; Some system services need to execute some form
; of synchronization code before returning back to
; the caller.  The following code changes the
; return PC placed on the stack by CHMK
; to be the address of the synchronization code
; to be executed.
;
; CMOD$AL_EXIT_TYPE - Table containing the addresses of
;                     synchronization code indexed by
;                     the contants of DISP_B_EXIT_TYPE(R1)
;
                                                    ;
                    MOVZBL  DISP_B_EXIT_TYPE(R1),R2         ; Get the index into
                                                            ; CMOD$AL_EXIT_TYPE
                    MOVL    CMOD$AL_EXIT_TYPE[R2],20(SP)    ; Change the return address
                                                            ;
; Most kernel mode system services need the PCB address. For that reason,
; the PCB address of the current process is loaded into R4 before the dispatch
```

```
628 ; takes place. Note that this requires all kernel mode services to include
629 ; R4 in their entry mask.
630
631 5$:     JMP     @DISP_A_SERVICE_ROUTINE(R1)
632
633 ; A CHMK instruction was executed with a code that does not represent an
634 ; existing service. Try the various loadable service dispatchers before
635 ; returning an error status.
636 ;
637 ; Note that these various dispatchers can either ACCEPT the service code
638 ; (and eventually return with a RET to SERVICE_EXIT) or reject the code. In
639 ; this latter case, they return here with an RSB to allow the next dispatcher
640 ; in line to be tried.
641
642 10$:    TSTB    G^CTL$GB_SSFILTER
643         BNEQ    40$
644
645         MOVL    G^CTL$GL_USRCHMK, R1    ;
646         BEQL    20$
647         JSB     (R1)
648
649 20$:    MOVL    G^EXE$GL_USRCHMK, R1    ;
650         BEQL    30$
651         JSB     (R1)
652
653 30$:    MOVZWL  #SS$_ILLSER, R0
654         RET
655
656 40$:    MOVZWL  #SS$_INHCHMK, R1        ; Load failure code
657         BRW     INHEXCP1                ; Reflect the error
```

```
659          .SUBTITLE       Error Handling for KERNEL Mode Services
660 ;+
661 ; Functional Description:
662 ;
663 ;       These routine receive control when the change mode dispatcher delects
664 ;       an error while checking the validity of the argument list.
665 ;
666 ; Input Parameters:
667 ;
668 ;       Identical to the output of the change mode dispatcher
669 ;
670 ; Implicit Output:
671 ;
672 ;       If the change mode code is recognized, then the service is immediately
673 ;       dismissed with an appropriate status. Note that errors detected by the
674 ;       dispatcher are not signalled as system service failure exceptions.
675 ;-
676
677 KERNEL_ACCVIO:
678          MOVZWL  #SS$_ACCVIO, R0
679          RET
680
681 KERNEL_INSFARG:
682          MOVZWL  #SS$_INSFARG, R0
683          RET
684
685          .DISABLE        LOCAL_BLOCK
```

```
                .SUBTITLE       SERVICE_EXIT - System Service Completion
;+
; Functional Description:
;
;       This code receives control on return from the service-specific
;       procedures. The success path is a simple exit (with an REI) back to
;       the location of the CHMx instruction. If the service is reporting an
;       error, this routine decides whether to merely report that error back
;       to the caller or to signal an exception (SS$_SSFAIL).
;
; Input Parameters:
;
;       R0 - Final status from service-specific procedure
;
; Implicit Input:
;
;       PCB$V_SSFEXC flag for caller's mode
;
; Side Effects:
;
;       If a system service has not released all owned mutexes, this
;       routine issues a fatal bugcheck.
;-

SERVICE_EXIT::
        BLBC    R0, 20$                 ; Branch if error
10$:    REI

20$:    BITL    #7, R0                  ; Was error merely a warning?
        BEQL    10$                     ; No exception in that case


; Make sure that the service did not forget to unlock mutexes along
; some error path.

        MOVL    G^CTL$GL_PCB, R1        ; Get PCB address
        TSTW    PCB$W_MTXCNT(R1)        ; Any mutexes?
        BNEQ    40$

; See if system service failure exceptions are enabled for caller's mode

        EXTZV   #PSL$V_CURMOD, -        ; Get caller's mode
                #PSL$S_CURMOD, -        ; ... from saved PSL
                4(SP), -(SP)
        ADDL    #PCB$V_SSFEXC, (SP)     ; Add in base bit number
        BBC     (SP)+, PCB$L_STS(R1), - ; If bit is clear, system service
                10$                     ;  failure exceptions are disabled

; Before we pass control to the exception dispatcher, let us make sure
; that IPL is zero. Exception dispatching at other than IPL 0 can cause
; a variety of problems. Note that IPL can only be nonzero if the current
; mode is kernel.

        MOVPSL  -(SP)                   ; Get current PSL
        EXTZV   #PSL$V_CURMOD, -        ; If the current mode is other than
                #PSL$S_CURMOD, -        ;  kernel,
                (SP), (SP)+             ;
```

```
    SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 18
    X-12 SERVICE_EXIT - System Service Completion 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER

      744          BNEQ    30$                     ; ... then IPL is guaranteed 0.
      745          SETIPL  #0                      ; Force IPL to 0 for error path
      746 30$:     JMP     G^EXE$SSFAIL            ; Generate the exception
      747
      748 ; A system service returned control to its caller while still controlling
      749 ; mutexes. If the call was made from elevated IPL (IPL$ASTDEL or higher),
      750 ; then let this anamoly pass. Otherwise, issue a bugcheck.
      751
      752 40$:     EXTZV   #PSL$V_IPL, -           ; Extract caller's IPL
      753                  #PSL$S_IPL, -           ; ... from saved PSL
      754                  4(SP), -(SP)
      755          CMPL    (SP)+, #IPL$_ASTDEL     ; Test for elevated IPL
      756          BGEQ    10$                     ; Only bugcheck if at IPL 0
      757          BUG_CHECK MTXCNTNONZ, FATAL     ; MUTEX count nonzero at service exit
```

```
SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 19
X-12 INHEXCP - Inhibited CHMK or CHME code ha 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER

          .SUBTITLE       INHEXCP - Inhibited CHMK or CHME code handling
;+
; Functional Description:
;
;       When the ability to use specified system services is inhibited via the
;       $SETSSF system service, this routine receives control when an attempt
;       to execute an inhibited system service occurs.
;
;       INHEXCP is called when no stack frame cleanup is required. That is,
;       the change mode dispatcher has not yet built a call frame on the
;       stack.
;
;       INHEXCP1 is called when a call frame must be cleared from the stack.
;
; Input Parameters:
;
;       Entry at INHEXCP
;
;               R1 - Failure code (SS$_INHCHMK or SS$_INHCHME)
;
;               00(SP) - Change mode code
;               04(SP) - Saved PC of CHMx exception
;               08(SP) - Saved PSL of CHMx exception
;
;       Entry at INHEXCP1
;
;               R0 - Change mode code
;               R1 - Failure code (SS$_INHCHMK or SS$_INHCHME)
;
;               00(SP) \
;               04(SP)  \
;               08(SP)   > Call frame to be disacrded
;               12(SP)  /
;               16(SP) /
;               20(SP) - Saved PC of exception
;               24(SP) - Saved PSL of exception
;
; Output Parameters:
;
;       none
;
; Implicit Output:
;
;       An exception is generated with the following signal array
;
;               Argument Count (always 4)
;               Signal Name (SS$_INHCHMK or SS$_INHCHME)
;               Inhibited change mode code whose use was attempted
;               PC following CHMx instruction
;               PSL of CHMx instruction
;-

INHEXCP1:
        MOVL    12(SP),FP               ;PICK UP THE OLD FP FROM FRAME
        ADDL    #5*4,SP                 ;CLEAN OFF THE FRAME
        PUSHL   R0                      ;RESTORE THE CHMX CODE
INHEXCP:
```

```
    SYSTEM_SERVICE_DISPATCHER - Change Mode System Service Dispatcher 10-MAY-1989 16:08:15 VAX M
8 Page 20
    X-12 INHEXCP - Inhibited CHMK or CHME code ha 23-JUN-1988 14:57:50 SYSTEM_SERVICE_DISPATCHER

      816          PUSHL   R1                      ;PUSH THE EXCEPTION CODE
      817          PUSHL   #4                      ;PUSH THE NUMBER OF ARGUMENTS
      818          JMP     G^EXE$REFLECT           ;REFLECT THE EXCEPTION
      819
      820          .END
```

10 SYSTEM_SERVICE_EXIT.LIS

```
    SYSTEM_SERVICE_EXIT - System service exit/synchronization co 10-MAY-1989 16:09:08 VAX MACRO
8 Page 0
    Table of contents

      (2)       106         SYNCH$RMS_STALL - I/O stall synchronization
      (3)       196         SYNCH$RMS_WAIT - Synchronization code for RMS$WAIT
      (4)       243         SYNCH$ASSIGN_EXIT - Special check on exit from $ASSIGN service
```

```
    SYSTEM_SERVICE_EXIT - System service exit/synchronization co 10-MAY-1989 16:09:08 VAX MACRO
8 Page 1
    X-5 28-AUG-1987 18:02:17 SYSTEM_SERVICE_EXIT.MAR;1 (1)

     1          .TITLE  SYSTEM_SERVICE_EXIT - System service exit/synchronization code
     2          .IDENT  'X-5'
     3 ;
     4 ;*****************************************************************************
     5 ;*                                                                           *
     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
     8 ;*  ALL RIGHTS RESERVED.                                                     *
     9 ;*                                                                           *
    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
    11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
    15 ;*  TRANSFERRED.                                                             *
    16 ;*                                                                           *
    17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
    18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
    19 ;*  CORPORATION.                                                             *
    20 ;*                                                                           *
    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
    23 ;*                                                                           *
    24 ;*                                                                           *
    25 ;*****************************************************************************
    26 ;
    27 ; Facility:
    28 ;
    29 ;       VAX/VMS Executive, EXCEPTION image.
    30 ;
    31 ; Abstract:
    32 ;
    33 ;       Code contained in this module, previously contained
    34 ;       in the now dead CMODSSDSP.MAR module.  Yeah.
    35 ;
    36 ;       The code was formerly assembled into system service
    37 ;       vector area, and was gotton to by
    38 ;       branches within the RMS system service vectors.
    39 ;       Now, this code is gotton to by changing the return
    40 ;       PC on the system service call frame by the
    41 ;       system service dispatcher.
    42 ;
    43 ; Author:
    44 ;
    45 ; D. N. CUTLER 22-JUN-76
    46 ;
    47 ;  MODIFIED BY:
    48 ;
    49 ;       X-5     SAD0049         Stuart A. Davidson      27-AUG-1987
    50 ;               Add exit routine for new exit type (ASSIGN_EXIT).  It checks
    51 ;               for SS$_NONLOCAL, to execute $ASSIGN to network object in
    52 ;               caller's mode.
    53 ;
    54 ;       X-4     PJH             Paul J. Houlihan                26-May-1987
    55 ;               The $WAIT exit code, changed by Exec reorg, was formerly called
    56 ;               via JSB. The $WAIT exit code would then decide to either RSB or
    57 ;               continue executing the $WAIT exit code by popping and discarding
```

```
                ;               the return PC from the JSB. The new exec reorg mechanism is to
                ;               fudge the system service call frame with the PC of the correct
                ;               exit routine, thus no JSB is involved. The problem is that the
                ;               code is still discarding the (now non-existent) return PC.
                ;
                ;       X-3     SF04003         Stephen Fiorelli                09-Apr-1986
                ;               Add the call to RMS$SSVECX in case we need to generate .
                ;               a possible system service failure exception.
                ;
                ;       X-2     SF04002         Stephen Fiorelli                11-Jan-1986
                ;               Change EXE$WAIT to RMS$WAIT
                ;
                ;       X-1     SF04001         Stephen Fiorelli                17-Dec-1985
                ;               Moved the code to this new module, and cleaned it up.
                ;
                ;+


                ;
                ;       Macro library calls
                ;

                $RABDEF                         ; Define RMS RAB fields
                $RMSDEF                         ; RMS definitions
                $SSDEF                          ; Define system status values
                $STSDEF                         ; Define system status values

                ;
                ;       External references
                ;

                .EXTERNAL       RMS$WAIT                ; RMS $WAIT vector
                .EXTERNAL       CLREF                   ; CLREF change mode code
                .EXTERNAL       SETEF                   ; SETEF change mode code
                .EXTERNAL       WAITFR                  ; WAITFR change mode code
                .EXTERNAL       EXE$NETWORK_ASSIGN      ; Caller's mode entry point
                                                        ; for network $ASSIGN

                ;
                ;       Special CHME code to generate a possible system
                ;       service failure exception due to an error occuring
                ;       in the rms synchronization code.
                ;

SSVECX=0

                .DISABLE        GLOBAL
                DECLARE_PSECT   EXEC$NONPAGED_CODE
```

```
    SYSTEM_SERVICE_EXIT - System service exit/synchronization co 10-MAY-1989 16:09:08 VAX MACRO
8 Page 3
    X-5 SYNCH$RMS_STALL - I/O stall synchronizat 28-AUG-1987 18:02:17 SYSTEM_SERVICE_EXIT.MAR;1

      106          .SUBTITLE       SYNCH$RMS_STALL - I/O stall synchronization
      107 ;+
      108 ;  This is the synchronization code executed after an RMS
      109 ;  system service is executed if there is to be a wait for
      110 ;  I/O completion.
      111 ;
      112 ;  Note that most rms service vector terminates with a branch to this
      113 ;  routine.
      114 ;
      115 ;  The FAB/RAB is checked for a legal block id, i.e., a 1 or 3, and
      116 ;  an error returned if invalid.  The structure is not reprobed.
      117 ;
      118 ;  This routine assumes that the following registers have been set by the
      119 ;  exiting rms exec-level code whenever a stall is required:
      120 ;
      121 ;       R3      EFN to wait on
      122 ;       R8      RAB/FAB address to wait on
      123 ;
      124 ;-
      125
      126 SYNCH$RMS_STALL::
      127          CMPW    R0,#RMS$_STALL&^XFFFF   ; Is the status code I/O stall?
      128          BEQL    RMSWAIT_IO_DONE         ; Branch if yes
      129          RET                             ; Back to caller
      130
      131 ;
      132 ;  Set a flag in the user's control block that tells RMS that the process
      133 ;  is waiting on this FAB/RAB.  When RMS is initializing for a new operation
      134 ;  it checks this flag and rejects the new operation if the control block
      135 ;  is waiting on a previous operation.  This prevents a hang condition
      136 ;  caused by using the same STS/STV field for 2 operations at once.
      137 ;  FAB$B_BLN = RAB$B_BLN
      138 ;
      139
      140 RMSWAIT_IO_DONE:
      141          BISB    #1,RAB$B_BLN(R8)        ; Low bit of BLN field is the flag
      142
      143 ;
      144 ;  The arguments are pushed on the stack and the AP set up as if a 'CALLS'
      145 ;  instruction were being executed.  The change mode to kernel service is
      146 ;  executed directly.  This saves the overhead of a 'CALLS' instruction.
      147 ;  R8 must not be destroyed by any of the services used here.
      148 ;
      149          PUSHL   R3                      ; Event flag to wait for
      150          MOVAB   -4(SP),AP               ; Set up AP as if using CALLS instr
      151          PUSHL   #1                      ; Number of arguments
      152 USERWAIT:
      153          CHMK    W^WAITFR                ; Do 'naked' WAITFR to save calls time
      154
      155 ;
      156 ;  Check to see if the user structure pointed to by R8 is still valid by
      157 ;  checking the block id to be sure that it is either a RAB (BID=1) or
      158 ;  a FAB (BID=3).  This won't catch the case where what should have been
      159 ;  a FAB now looks like a RAB or vice versa  but will catch everything
      160 ;  else.  If the structure is not readable or writeable then the user
      161 ;  will get an access violation.  the BID for a FAB/RAB is at byte 0,
      162 ;  the STS for a FAB/RAB is at byte 8.
```

```
    SYSTEM_SERVICE_EXIT - System service exit/synchronization co 10-MAY-1989 16:09:08 VAX MACRO
8 Page 4
    X-5 SYNCH$RMS_STALL - I/O stall synchronizat 28-AUG-1987 18:02:17 SYSTEM_SERVICE_EXIT.MAR;1

      163 ;
      164
      165 10$:    BLBC    (R8),30$                ; Not set, then not a FAB or RAB
      166         BITB    #^B11111100,(R8)        ; Is it a 1 or 3?
      167         BNEQ    30$                     ; NEQ no so blow the whistle
      168         MOVL    8(R8),R0                ; Get RMS status code
      169         BEQL    20$                     ; and wait again if not set
      170         BICB    #1,RAB$B_BLN(R8)        ; Clear waiting flag
      171         BLBC    R0,30$                  ; Branch if failure code
      172         RET                             ; Return to caller
      173
      174 ;
      175 ;  Clear the RMS event flag, check status again and wait 1 more time if
      176 ;  operation still not done.  The appropriate arguments for the CLREF
      177 ;  and SETEF (if executed) remain on the stack from the WAITFR above.
      178 ;  The AP must be preserved.
      179 ;
      180
      181 20$:    CHMK    W^CLREF                 ; Do a 'naked' CLREF, the arguments
      182                                         ; are on stack and AP still set up
      183                                         ; from the WAITFR above
      184         TSTL    8(R8)                   ; And re-check status
      185         BEQL    USERWAIT                ; Branch to wait for flag again..
      186                                         ; ... if status still zero
      187         CHMK    W^SETEF                 ; I/O complete - leave EFN set
      188         BRB     10$                     ; and restore r0 status code
      189 ;
      190 ;  Branch to check status code for error or severe error
      191 ;  a success status in R0 (from the $WAITFR) indicates an invalid FAB/RAB.
      192 ;
      193
      194 30$:    BRW     RMS_ERR
```

```
    SYSTEM_SERVICE_EXIT - System service exit/synchronization co 10-MAY-1989 16:09:08 VAX MACRO
8 Page 5
    X-5 SYNCH$RMS_WAIT - Synchronization code fo 28-AUG-1987 18:02:17 SYSTEM_SERVICE_EXIT.MAR;1

      196          .SUBTITLE       SYNCH$RMS_WAIT - Synchronization code for RMS$WAIT
      197 ;
      198 ;
      199 ;  RMS $WAIT SYNCHRONIZATION CODE.
      200 ;
      201 ;  Look at flag in R4 to determine if this is a $WAIT for the same or different
      202 ;  RABS.  If same, merely RSB;  if different, wait on event flag and then
      203 ;  re-execute the $WAIT service.
      204 ;       R4      flag for wait type
      205 ;               (0 = same RAB, 1 = different RABS)
      206 ;
      207
      208 SYNCH$RMS_WAIT::
      209          BLBS    R4,10$                  ; Branch if different rabs
      210          BRW     SYNCH$RMS_STALL         ; Handle with standard stall
      211 10$:     CMPW    R0,#RMS$_STALL&^XFFFF   ; Is stall required?
      212          BEQL    20$                     ; Branch if yes
      213          RET                             ; No - back to user
      214 20$:     $WAITFR_S       R3              ; Wait on specified event flag
      215          JMP     G^RMS$WAIT+2            ; Re-execute RMS $WAIT
      216
      217 ;
      218 ;  The following code is an error path from the rms sychronization code.
      219 ;
      220 ;  Check status code for error or severe error, if success then
      221 ;  bad user structure detected - return error in R0, status of record
      222 ;  operation will be lost
      223 ;
      224
      225 RMS_ERR:
      226          BICB2   #1,RAB$B_BLN(R8)        ; Clear waiting flag
      227          BLBC    R0,98$                  ; Stale success => bad structure
      228          MOVL    #RMS$_STR,R0            ; Change status to bad structure error
      229 98$:     BITB    #6,R0                   ; Error or severe error?
      230          BEQL    99$                     ; Branch if not
      231
      232 ;
      233 ;  Must return to exec mode to generate possible system
      234 ;  service failure exception.
      235 ;
      236
      237          MOVL    R0,R2                   ; Status code to R2
      238          CHME    I^#SSVECX               ; Generate exception if enabled
      239
      240 99$:     RET
      241
```

```
SYSTEM_SERVICE_EXIT - System service exit/synchronization co 10-MAY-1989 16:09:08 VAX MACRO
8 Page 6
X-5 SYNCH$ASSIGN_EXIT - Special check on exi 28-AUG-1987 18:02:17 SYSTEM_SERVICE_EXIT.MAR;1

                .SUBTITLE SYNCH$ASSIGN_EXIT - Special check on exit from $ASSIGN service
;
;
;  $ASSIGN - check for network object assign operation
;
;  Examine the completion status to see if we should attempt a network
;  assign.  If the status is not SS$_NONLOCAL, just return to user.

SYNCH$ASSIGN_EXIT::
        CMPW    R0,#SS$_NONLOCAL!STS$K_SUCCESS
                                        ; Test for $ASSIGN to a network
                                        ;  object.
        BEQL    5$                      ; Branch on match
        RET                             ; Non-network, just return
5$:     JMP     G^EXE$NETWORK_ASSIGN    ; Return to SYSASSIGN for completion
                                        ; now in callers mode.


        .END
```

11 SYSTEM_SERVICE_LOADER.LIS

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 0
    Table of contents

      (2)       93        Declarations
      (3)      173        EXEC$CONNECT_SYSTEM_SERVICES - Add a Service to the Dispatcher
      (4)      234        Load CALLERS_MODE system service
      (5)      265        Load KERNEL or EXEC system services (consistency checking)
      (6)      305        Load KERNEL and EXEC system services (Load the vector in base image)
      (7)      350        Load KERNEL and EXEC system services (Load the auxiliary arrays)
      (8)      421        Reload a system service
      (9)      494        Composite service loading
      (10)     550        LOCK/UNLOCK - Control Access to CHMx Codes
```

```
SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 1
X-10 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAR;1 (1)

     1          .TITLE  SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher
     2          .IDENT  'X-10'
     3
     4  ;*****************************************************************************
     5  ;*                                                                           *
     6  ;*  COPYRIGHT (c) 1985 BY                                                    *
     7  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
     8  ;*  ALL RIGHTS RESERVED.                                                     *
     9  ;*                                                                           *
    10  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
    11  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
    12  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
    13  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
    14  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
    15  ;*  TRANSFERRED.                                                             *
    16  ;*                                                                           *
    17  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
    18  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
    19  ;*  CORPORATION.                                                             *
    20  ;*                                                                           *
    21  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
    22  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
    23  ;*                                                                           *
    24  ;*                                                                           *
    25  ;*****************************************************************************
    26  ;++
    27  ; Facility:
    28  ;
    29  ;       VAX/VMS Executive, EXCEPTION image
    30  ;
    31  ;       System Service Dispatcher for Kernel and Exec Access Modes
    32  ;
    33  ; Abstract:
    34  ;
    35  ;       The routines in this module connect system services that have been
    36  ;       loaded into system space into the appropriate change mode dispatcher.
    37  ;       Service-specific entries are added to the dispatch vector and a change
    38  ;       mode code is allocated for each system service in each loadable image.
    39  ;
    40  ; Author:
    41  ;
    42  ;       Lawrence J. Kenah
    43  ;
    44  ; Creation Date:
    45  ;
    46  ;       20 May 1985
    47  ;
    48  ; Modified by:
    49  ;
    50  ;       X-10    SF04007         Stephen Fiorelli        23-Feb-1987
    51  ;               Make this routine pageable.
    52  ;
    53  ;       X-9     SF04006         Stephen Fiorelli        27-Jan-1987
    54  ;               Also or in R4 into the register masks of all system
    55  ;               system services.
    56  ;
    57  ;       X-8     SF04005         Stephen Fiorelli        10-Sep-1986
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 2
    X-10 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAR;1 (1)

       58 ;                Remove fatal status codes, bugcheck instead.
       59 ;
       60 ;        X-5,6,7 SF04004         Stephen Fiorelli        03-Feb-1986
       61 ;                Change name of main routine from EXE$CONNECT_SYSTEM_SERVICES
       62 ;                to EXE$CONNECT_SERVICES.  Eliminate references to change protection
       63 ;                routines.
       64 ;
       65 ;        X-4     SF04003         Stephen Fiorelli        17-Dec-1985
       66 ;                Load a new field of the system service dispatcher
       67 ;                record (CMOD$AR_KERNEL_VECTOR or CMOD$AR_EXEC_VECTOR).
       68 ;                This new field will contain an index into the table
       69 ;                CMOD$AL_EXIT_TYPE, which will contain the address of
       70 ;                exit code other than default (index=0),
       71 ;                to be executed on return from the system service.
       72 ;                This field will be used by the system service dispatcher
       73 ;                when building a call frame for the system specific
       74 ;                procedure.  If this new field does not contain a
       75 ;                zero (the default), the return address is changed
       76 ;                to be the address of the contants of CMOD$AL_EXIT_TYPE
       77 ;                indexed by the field.
       78 ;
       79 ;                Also R2 ORed into the register masks of all system
       80 ;                services in the vector area.  This was done because
       81 ;                R2 was needed for use in the dispatcher.
       82 ;
       83 ;        X-3     SF04002         Stephen Fiorelli        15-Nov-1985
       84 ;                Add support for composite system service loading.
       85 ;
       86 ;        X-2     SF04001         Stephen Fiorelli        16-Oct-1985
       87 ;                Eliminate jacket routine, change PSECT names.
       88 ;                Add Universal Entry macro.  Fix miscellaneous bugs.
       89 ;
       90 ;        X-1     LJK4007         Lawrence J. Kenah       Original
       91 ;                The existing system services are added to the system.
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 3
    X-10 Declarations 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAR;1 (2)

      93          .SUBTITLE       Declarations
      94
      95 ; Include files:
      96
      97          $OPDEF                          ; Symbolic representation of opcodes
      98
      99          $DISPDEF                        ; System service dispatch vector
     100          $SSDESCRDEF                     ; System service descriptor block
     101          $SSVECDEF                       ; System service vector
     102
     103 ; External declarations
     104
     105          .DISABLE        GLOBAL
     106
     107 ; External cells
     108
     109 ; External routines
     110
     111          .EXTERNAL       -
     112                  SCH$LOCKW, -
     113                  SCH$UNLOCK, -
     114                  SMP$GET_CURPCB
     115
     116 ; Global data cells
     117
     118          .EXTERNAL       -               ; ... for exec mode services
     119                  CMOD$GW_CHME_LIMIT, -
     120                  CMOD$AR_EXEC_DISPATCH_VECTOR, -
     121                  CMOD$AB_EXEC_INHIBIT_MASK, -
     122                  CMOD$AB_EXEC_MIN_ARG_COUNT
     123
     124          .EXTERNAL       -               ; ... and for kernel mode services
     125                  CMOD$GW_CHMK_LIMIT, -
     126                  CMOD$AR_KERNEL_DISPATCH_VECTOR, -
     127                  CMOD$AB_KERNEL_INHIBIT_MASK, -
     128                  CMOD$AB_KERNEL_MIN_ARG_COUNT
     129
     130
     131          .EXTERNAL       -               ; composite system service tables
     132                  CMOD$AL_COMPOSITE_VECTOR,-
     133                  CMOD$AL_COMPOSITE_CODE,-
     134                  CMOD$AW_COMPOSITE_LENGTH
     135
     136          .EXTERNAL       -
     137                  CHANGE_MODE_MUTEX
     138
     139 ; Status codes
     140
     141          .EXTERNAL       -
     142                  SS$_NORMAL
     143
     144 ; Bugcheck codes
     145
     146          .EXTERNAL       -
     147                  BUG$_BADVECTOR,-
     148                  BUG$_SSVECFULL
     149
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 4
    X-10 Declarations 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAR;1 (2)

      150 ; Local equates
      151
      152          IMMEDIATE_MODE = ^X8F
      153          ABSOLUTE_MODE = ^X9F
      154
      155
      156 ; Local macros
      157
      158          .MACRO  LOCK
      159          BSBW    LOCK_CHMX_CODE
      160          .ENDM   LOCK
      161
      162          .MACRO  UNLOCK
      163          BSBW    UNLOCK_CHMX_CODE
      164          .ENDM   UNLOCK
      165
      166 ; PSECT Declarations:
      167
      168          .DEFAULT        DISPLACEMENT , WORD
      169
      170          DECLARE_PSECT   EXEC$PAGED_CODE
      171
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 5
    X-10 EXEC$CONNECT_SYSTEM_SERVICES - Add a Ser 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAF

      173          .SUBTITLE       EXEC$CONNECT_SYSTEM_SERVICES - Add a Service to the Dispatch
      174 ;+
      175 ; Functional Description:
      176 ;
      177 ;       This routine adds a system service to the dispatch vector and
      178 ;       allocates a change mode code for the service.
      179 ;
      180 ; Input Parameters:
      181 ;
      182 ;       4(AP) - Address of system service descriptor block in loadable image
      183 ;
      184 ; Implicit Input:
      185 ;
      186 ;       Contents of system service vector for this service
      187 ;
      188 ;       Contents of dispatch vector array and other arrays used by the
      189 ;       change mode dispatcher.
      190 ;
      191 ; Output Parameters:
      192 ;
      193 ;       none
      194 ;
      195 ; Implicit Output:
      196 ;
      197 ;       If this service executes in exec or kernel mode, several changes are
      198 ;       made
      199 ;
      200 ;               A change mode code (exec or kernel) is allocated.
      201 ;
      202 ;               An entry is made in the appropriate dispatch vector (exec
      203 ;               or kernel) and the other vectors used by the change mode
      204 ;               dispatcher.
      205 ;
      206 ;               The contents of the system service vector are changed to
      207 ;               a CHMx instruction with the new change mode code followed
      208 ;               by a RET instruction.
      209 ;-
      210
      211          .ENABLE         LOCAL_BLOCK
      212
      213          UNIVERSAL_ENTRY EXE$CONNECT_SERVICES, <^M<R2,R3,R4,R5>>
      214
      215          MOVL    4(AP),R4                ; R4 points to descriptor block
      216          MOVL    SSDESCR_A_VECTOR_ADDRESS(R4), -
      217                  R2                      ; R2 points to system service vector
      218          ADDL3   #2, -                   ; R3 points to first instruction
      219                  SSDESCR_A_ENTRY_ADDRESS(R4), -
      220                  R3                      ; ... of service-specific procedure
      221
      222          CASE    SSDESCR_B_MODE(R4), TYPE=B, LIMIT=#MODE_K_KERNEL, -
      223                  <-
      224                  KERNEL, -               ; Service executes in KERNEL mode
      225                  EXEC, -                 ; Service executes in EXEC mode
      226                  10$, -                  ; Error if mode is supervisor
      227                  10$, -                  ; ... or user
      228                  CALLERS_MODE -          ; Service executes in caller's mode
      229                  >
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 6
    X-10 EXEC$CONNECT_SYSTEM_SERVICES - Add a Ser 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAR

      230
      231 10$:    BUG_CHECK BADVECTOR,FATAL       ; Illegal mode
      232         RET
```

```
SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 7
X-10 Load CALLERS_MODE system service 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAR;1 (4)

                .SUBTITLE       Load CALLERS_MODE system service

; Service executes in caller's mode. Make sure that the contents of the
; system service vector contain a JMP instruction to the correct service
; routine with the correct register save mask.

CALLERS_MODE:
        CMPW    SSVEC_W_ENTRY_MASK(R2), -
                -2(R3)                  ; Do entry masks agree?
        BNEQ    20$                     ; Error if not
        CMPB    SSVEC_B_OPCODE(R2), -
                #OP$_JMP                ; JMP instruction?
        BNEQ    20$
        CMPB    SSVEC_B_ADDRESSING_MODE(R2), -
                #ABSOLUTE_MODE          ; JMP @# ?
        BNEQ    20$
        CMPL    SSVEC_A_JMP_DESTINATION(R2), -
                R3                      ; Does it go to the right place?
        BNEQ    20$

; The system service is consistent with the system service descriptor block

        BRW     NORMAL_EXIT             ; Normal exit code, a check for a
                                        ; composite service will be done

; An inconsistency was discovered

20$:    BUG_CHECK BADVECTOR,FATAL       ; Inconsistent vector status
        RET
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 8
    X-10 Load KERNEL or EXEC system services (con 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAF

      265          .SUBTITLE       Load KERNEL or EXEC system services (consistency checking)
      266
      267 ; The service execues in EXEC or KERNEL mode. If the system service vector
      268 ; contains a JMP instruction to the service-specific procedure, we are
      269 ; loading the service for the first time. If the vector contains a CHMx
      270 ; instruction, we are reloading the service.
      271
      272 KERNEL:
      273          LOCK
      274          MOVZWL  CMOD$GW_CHMK_LIMIT, R0  ; R0 gets next change mode code
      275          MOVZBL  #OP$_CHMK, R1           ; R1 contains mode-specific opcode
      276          BRB     30$
      277
      278 EXEC:
      279          LOCK
      280          MOVZWL  CMOD$GW_CHME_LIMIT, R0  ; R0 gets next change mode code
      281          MOVZBL  #OP$_CHME, R1           ; R1 contains mode-specific opcode
      282
      283 30$:     CMPB    SSVEC_B_OPCODE(R2), -
      284                  #OP$_JMP                ; JMP instruction?
      285          BEQL    35$                     ; It is. Load the service
      286          BRW     RELOAD                  ; If not, see if this is a reload
      287
      288 35$:     CMPW    SSVEC_W_ENTRY_MASK(R2), -
      289                  -2(R3)                  ; Do entry masks agree?
      290          BNEQ    40$                     ; Error if not
      291          CMPB    SSVEC_B_ADDRESSING_MODE(R2), -
      292                  #ABSOLUTE_MODE          ; JMP @# ?
      293          BNEQ    40$
      294          CMPL    SSVEC_A_JMP_DESTINATION(R2), -
      295                  R3                      ; Does it go to the right place?
      296          BEQL    LOAD
      297
      298 ; Some sort of inconsistency was detected. Release the interlock and
      299 ; report the error.
      300
      301 40$:
      302          UNLOCK
      303          BRB     20$
```

```
SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 9
X-10 Load KERNEL and EXEC system services (Lo 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAF

        .SUBTITLE       Load KERNEL and EXEC system services (Load the vector in bas

; The system service vector contains a JMP instruction to the service-specific
; procedure, indicating the initial load of the service.

LOAD:
        CMPW    R0, #256                ; Too many services already loaded?
        BLSSU   45$                     ; Report error if yes
        BUG_CHECK SSVECFULL,FATAL
        RET

; Store the revised instruction sequence in the system service vector. This
; is accomplished in two steps. The instruction sequence is assembled on the
; top of the stack and moved as a single unit to the system service vector.
; This prevents other processes from attempting to execute a hybrid
; instruction sequence.

        ASSUME SSVEC_K_LENGTH EQ 8      ; Make sure it's a quadword
45$:
        SUBL    #SSVEC_K_LENGTH, SP     ; Allocate space on the stack

        MOVW    SSVEC_W_ENTRY_MASK(R2), -
                SSVEC_W_ENTRY_MASK(SP)  ; Make a copy of the entry mask
        BISW2   #^M<R2,R4>,-            ; OR in register 2 and 4 used by
                SSVEC_W_ENTRY_MASK(SP)  ; system service dispatcher
                                        ; by system service dispatcher
        MOVB    R1, SSVEC_B_OPCODE(SP)  ; CHMx instruction
        MOVB    #IMMEDIATE_MODE, -      ; ... with word immediate mode (I^#)
                SSVEC_B_ADDRESSING_MODE(SP)
        MOVW    R0, -                   ; Fill in change mode code
                SSVEC_W_CHMX_CODE(SP)
        MOVZBW  #OP$_RET, -
                SSVEC_B_RET(SP)         ; ... followed by a RET instruction

; Move the entire sequence to the system service vector and execute an REI
; instruction to purge an instruction buffer.

        MOVQ    (SP)+, (R2)             ; Store the instruction sequence
        MOVPSL  -(SP)                   ; Get the current PSL
        BSBW    80$                     ; Execute an REI instruction

        TSTB    SSDESCR_B_MODE(R4)      ; Kernel mode service?
        BNEQ    50$                     ; Branch if exec mode
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 10
    X-10 Load KERNEL and EXEC system services (Lo 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAF

      350         .SUBTITLE       Load KERNEL and EXEC system services (Load the auxiliary arr
      351
      352 ; KERNEL mode service
      353 ;
      354 ; Fill in the auxiliary arrays from the system service descriptor BLOCK
      355
      356         MOVB    SSDESCR_B_MIN_ARG_COUNT(R4), -
      357                 CMOD$AB_KERNEL_MIN_ARG_COUNT[R0]
      358         MOVB    SSDESCR_B_INHIBIT_MASK(R4), -
      359                 CMOD$AB_KERNEL_INHIBIT_MASK[R0]
      360         MOVAQ   CMOD$AR_KERNEL_DISPATCH_VECTOR[R0], -
      361                 R0                      ; R0 points to next dispatch vector
      362         MOVAW   CMOD$GW_CHMK_LIMIT, R2  ; Store the address of the CHMK counter
      363         BRB     60$
      364
      365 ; EXEC mode service
      366 ;
      367 ; Fill in the auxiliary arrays from the system service descriptor block
      368 50$:
      369         MOVB    SSDESCR_B_MIN_ARG_COUNT(R4), -
      370                 CMOD$AB_EXEC_MIN_ARG_COUNT[R0]
      371         MOVB    SSDESCR_B_INHIBIT_MASK(R4), -
      372                 CMOD$AB_EXEC_INHIBIT_MASK[R0]
      373         MOVAQ   CMOD$AR_EXEC_DISPATCH_VECTOR[R0], -
      374                 R0                      ; R0 points to next dispatch vector
      375         MOVAW   CMOD$GW_CHME_LIMIT, R2  ; Store the address of the CHME counter
      376
      377 ; The routine can complete by executing code that is common for kernel and exec
      378 ; modes. (R1 ia a scratch register for the remainder of this routine.)
      379
      380 60$:    MOVZBL  SSDESCR_B_EXIT_TYPE(R4),-
      381                 R1                      ; Get exit type
      382         MOVB    R1, -                   ; Store it
      383                 DISP_B_EXIT_TYPE(R0)
      384         MOVZBL  SSDESCR_B_ARG_COUNT(R4), -
      385                 R1                      ; Get argument count
      386         MOVB    R1, -                   ; Store it
      387                 DISP_B_ARGUMENT_COUNT(R0)
      388
      389 ;temp till loader debugged
      390
      391 ; Calculate size in bytes of associated argument list
      392 ;
      393 ;       SIZE = 4 * (COUNT + 1)
      394
      395         INCL    R1
      396         MULW3   #4, R1, -               ; Store the size
      397                 DISP_W_ARG_LIST_SIZE(R0)
      398         MOVL    R3, -                   ; Load the JMP vector
      399                 DISP_A_SERVICE_ROUTINE(R0)
      400         INCW    (R2)                    ; Turn the service on
      401
      402 ; The service has been loaded successfully. Release the interlock, signal
      403 ; success, and return.
      404
      405 70$:
      406         UNLOCK
```

```
407          BRW     NORMAL_EXIT                ; Normal exit code, a check for a
408                                             ; composite service will be done
409
410 ; The following instruction is executed to invalidate the contents of any
411 ; instruction cache that might be implemented.
412
413 80$:     REI
414
415 ; The limit on the number of services that can be loaded has been reached.
416
417
418 95$:     BRW     40$                        ; Branch aid
419
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 12
    X-10 Reload a system service 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAR;1 (8)

      421          .SUBTITLE       Reload a system service
      422
      423 ; The service already exists (has a change mode code allocated). All that
      424 ; is needed is to reload the dispatch vector with the characteristics of
      425 ; the new service.
      426
      427 RELOAD:
      428          CMPB    SSVEC_B_OPCODE(R2), R1  ; Correct CHMx instruction
      429          BNEQ    95$                     ; Error if different
      430          CMPB    SSVEC_B_ADDRESSING_MODE(R2), -
      431                  #IMMEDIATE_MODE         ; Correct mode?
      432          BNEQ    95$
      433          CMPB    SSVEC_B_RET(R2), -
      434                  #OP$_RET                ; RET instruction in right place?
      435          BNEQ    95$
      436          MOVW    SSVEC_W_CHMX_CODE(R2), -
      437                  R0                      ; Pick up change mode code
      438
      439          TSTB    SSDESCR_B_MODE(R4)      ; Kernel mode service?
      440          BNEQ    100$                    ; Branch if exec mode
      441
      442 ; KERNEL mode service
      443 ;
      444 ; Fill in the auxiliary arrays from the system service descriptor block
      445
      446          MOVB    SSDESCR_B_MIN_ARG_COUNT(R4), -
      447                  CMOD$AB_KERNEL_MIN_ARG_COUNT[R0]
      448          MOVB    SSDESCR_B_INHIBIT_MASK(R4), -
      449                  CMOD$AB_KERNEL_INHIBIT_MASK[R0]
      450          MOVAQ   CMOD$AR_KERNEL_DISPATCH_VECTOR[R0], -
      451                  R0                      ; R0 points to next dispatch vector
      452          BRB     110$
      453
      454 ; EXEC mode service
      455 ;
      456 ; Fill in the auxiliary arrays from the system service descriptor block
      457
      458 100$:    MOVB    SSDESCR_B_MIN_ARG_COUNT(R4), -
      459                  CMOD$AB_EXEC_MIN_ARG_COUNT[R0]
      460          MOVB    SSDESCR_B_INHIBIT_MASK(R4), -
      461                  CMOD$AB_EXEC_INHIBIT_MASK[R0]
      462          MOVAQ   CMOD$AR_EXEC_DISPATCH_VECTOR[R0], -
      463                  R0                      ; R0 points to next dispatch vector
      464
      465 ; The routine can complete by executing code that is common for kernel and exec
      466 ; modes. (R1 ia a scratch register for the remainder of this routine.)
      467
      468 110$:    CMPB    SSDESCR_B_ARG_COUNT(R4), -
      469                  DISP_B_ARGUMENT_COUNT(R0)
      470          BEQL    120$
      471
      472 ; Argument list has changed size. New parameters must be calculated and stored.
      473
      474          MOVZBL  SSDESCR_B_ARG_COUNT(R4), -
      475                  R1                      ; Get argument count
      476          MOVB    R1, -                   ; Store it
      477                  DISP_B_ARGUMENT_COUNT(R0)
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 13
    X-10 Reload a system service 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAR;1 (8)

      478
      479 ; Calculate size in bytes of associated argument list
      480 ;
      481 ;       SIZE = 4 * (COUNT + 1)
      482
      483         INCL    R1
      484         MULW3   #4, R1, -               ; Store the size
      485                 DISP_W_ARG_LIST_SIZE(R0)
      486
      487 120$:   MOVL    R3, -                   ; Load the JMP vector
      488                 DISP_A_SERVICE_ROUTINE(R0)
      489
      490         BRB     80$                     ; Exit through common code
      491
      492         .DISABLE        LOCAL_BLOCK
```

```
    SYSTEM_SERVICE_LOADER - Connect Services to Dispatcher 10-MAY-1989 16:08:40 VAX MACRO V5.0-
8 Page 14
    X-10 Composite service loading 24-FEB-1987 08:48:18 SYSTEM_SERVICE_LOADER.MAR;1 (9)

      494          .SUBTITLE       Composite service loading
      495 ;+
      496 ; Functional Description:
      497 ;
      498 ;       Composite system service loading is done here.
      499 ;       After a successful loading of a system service
      500 ;       is complete, a check is made here to determine
      501 ;       if the system service has a wait form.
      502 ;
      503 ;+
      504
      505 NORMAL_EXIT:
      506          MOVZBL  SSDESCR_B_SYNCH_TYPE(R4),R2; Store the composite type
      507          BEQL    140$                    ; Branch if no composite type
      508
      509 ;
      510 ;       R2 ->   Composite type
      511 ;       R3 ->   Address of composite vector
      512 ;       R4 ->   System service descriptor block address for
      513 ;               non-wait form of the system service
      514 ;
      515 ;       Check if the vector has a JMP @#
      516 ;
      517
      518          MOVL    CMOD$AL_COMPOSITE_VECTOR[R2],R3 ; Get vector address
      519          CMPB    (R3),#OP$_JMP                   ; JMP instruction?
      520          BNEQ    150$                            ; No, error
      521          CMPB    1(R3),#ABSOLUTE_MODE            ; @#?
      522          BNEQ    150$                            ; No, error
      523
      524 ;
      525 ;       Now we know the service is in the vector area
      526 ;
      527 ;       R4 ->   Address of non-wait system service vector
      528 ;       R5 ->   Address of composite service code to be loaded
      529 ;
      530
      531          MOVL    CMOD$AL_COMPOSITE_CODE[R2],R5   ; Beginning of code area
      532          MOVL    SSDESCR_A_VECTOR_ADDRESS(R4),R4 ; Non-wait vector address
      533          CMPW    SSVEC_B_OPCODE(R4),-            ; Are chmx opcodes same?
      534                  SSVEC_B_OPCODE(R5)
      535          BNEQ    150$                            ; No, error
      536          TSTW    CMOD$AW_COMPOSITE_LENGTH[R2]    ; Does it have length?
      537          BEQL    150$                            ; No, error
      538          PUSHR   #^M<R0,R1,R2,R3,R4,R5>          ; Save some registers
      539          MOVC3   CMOD$AW_COMPOSITE_LENGTH[R2],-  ; Move the composite service
      540                  (R5),(R3)                       ; into the vector area
      541          POPR    #^M<R0,R1,R2,R3,R4,R5>          ; Restore some registers
      542          MOVW    SSVEC_W_CHMX_CODE(R4),-         ; Move in the change mode code
      543                  SSVEC_W_CHMX_CODE(R3)
      544 140$:    MOVZWL  #SS$_NORMAL, R0         ; Normal status
      545          RET                             ; Return
      546
      547 150$:    BUG_CHECK BADVECTOR,FATAL       ; Inconsistency found
      548          RET                             ; Return
```

```
550          .SUBTITLE       LOCK/UNLOCK - Control Access to CHMx Codes
551 ;+
552 ; Functional Description:
553 ;
554 ;       Only one process at a time can be adding system services to the system
555 ;       because access to OWN storage is required. This access is controlled
556 ;       by a mutex, locked for write access when a service is added. The two
557 ;       routines here merely serve as jackets for the mutex routines located
558 ;       elsewhere.
559 ;
560 ; Input Parameters:
561 ;
562 ;       none
563 ;
564 ; Implicit Input:
565 ;
566 ;       CHANGE_MODE_MUTEX - Semaphore that controls change mode codes
567 ;       CPU$L_CURPCB (of CPU-specific DB) - Address of PCB of calling process
568 ;
569 ; Output Parameters:
570 ;
571 ;       none
572 ;
573 ; Implicit Output:
574 ;
575 ;       Entry at LOCK_CHMX_CODE
576 ;
577 ;               The mutex is locked for write access
578 ;
579 ;       Entry at UNLOCK_CHMX_CODE
580 ;
581 ;               The mutex is released
582 ;
583 ; Side Effects:
584 ;
585 ;       R0 through R3 are destroyed as a result of calling these routines
586 ;
587 ;       R4 is preserved.
588 ;-
589
590 LOCK_CHMX_CODE:
591          PUSHL   R4
592          JSB     G^SMP$GET_CURPCB                ; Get current PCB address in R4
593          MOVAL   CHANGE_MODE_MUTEX, R0
594          JSB     G^SCH$LOCKW
595          POPL    R4
596          RSB
597
598 UNLOCK_CHMX_CODE:
599          PUSHL   R4
600          JSB     G^SMP$GET_CURPCB                ; Get current PCB address in R4
601          MOVAL   CHANGE_MODE_MUTEX, R0
602          JSB     G^SCH$UNLOCK
603          POPL    R4
604          RSB
605
606          .END
```

12 TIMESCHDL.LIS

```
TIMESCHDL - TIME DEPENDENT SCHEDULING 10-MAY-1989 16:56:06 VAX MACRO V5.0-8 Page 0
Table of contents
```

```
TIMESCHDL - TIME DEPENDENT SCHEDULING 10-MAY-1989 16:56:06 VAX MACRO V5.0-8 Page 1
X-44U3 1-NOV-1988 14:42:53 [SYS.SRC]TIMESCHDL.MAR;1 (1)

     1          .TITLE  TIMESCHDL - TIME DEPENDENT SCHEDULING
     2          .IDENT  'X-44U3'
     3
     4 ;
     5 ;*****************************************************************************
     6 ;*                                                                           *
     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
     9 ;*  ALL RIGHTS RESERVED.                                                     *
    10 ;*                                                                           *
    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
    16 ;*  TRANSFERRED.                                                             *
    17 ;*                                                                           *
    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
    20 ;*  CORPORATION.                                                             *
    21 ;*                                                                           *
    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
    24 ;*                                                                           *
    25 ;*                                                                           *
    26 ;*****************************************************************************
    27 ;
    28 ; D. N. CUTLER 15-JUN-76
    29 ;
    30 ; TIME RELATED ACTIVITY
    31 ;
    32 ;       UPDATE TIME OF DAY,
    33 ;       CHECK FOR ITEM READY IN TIMER QUEUE, AND
    34 ;       PERFORM DEVICE TIMEOUT.
    35 ;       UPDATE MEASUREMENT STATISTICS IF MEASUREMENT IS ENABLED.
    36 ;
    37 ; MODIFICATION HISTORY:
    38 ;
    39 ;       X-44U3  EMB0358         Ellen M. Batbouta       01-Nov-1988
    40 ;               Correctly separate busywait ticks that occur while
    41 ;               on the interrupt stack from other clock ticks and
    42 ;               correctly account for them.  Previously all time spent
    43 ;               on the interrupt stack, busywait or not, was attributed
    44 ;               to interrupt stack time.
    45 ;
    46 ;       X-44U2  SSA0002         Stan Amway              27-Oct-1988
    47 ;               Declare ASTWAIT resource available in EXE$TIMEOUT.
    48 ;
    49 ;       X-44U1  LPL001          Lee Leahy                9 Aug 1988
    50 ;               Changed reference in the comments to the symbol
    51 ;               EXE$AL_TQENOREPT to the version 5.0 symbol
    52 ;               EXE$AR_TQENOREPT.
    53 ;
    54 ;       X-44    SJF             Stu Farnham             11-Nov-1987
    55 ;               Check for halted CPUs before declaring CPUSANITY
    56 ;               bugcheck.
    57 ;
```

```
;       X-43    MSH0329         Michael S. Harvey       27-Oct-1987
;               Eliminate distinction of TIMKEEPER from PRIMARY CPU.
;
;       X-42    JWT0297         Jim Teague               6-Aug-1987
;               Use ADAWI for manipulating TQCNT.
;
;       X-41    SJF             Stu Farnham             28-Jul-1987
;               Correct sanity timer countdown.
;
;       X-40    WCT0076         Ward C. Travis          29-Apr-1987
;               Change old  occurrences of  UCB$B_ODIPL and  SMP$C_
;               to UCB$B_DIPL and SPL$C_, respectively.
;
;       X-39    RNG5039         Rod Gamache             23-Apr-1987
;               Add support for unmodified drivers - by checking the
;               UCB$B_FLCK field for a FIPL vs. FLCK.
;
;       X-38    SF00003         Stephen Fiorelli        15-Apr-1987
;               Set up pointer to permanent timer queue entry in
;               the base image.
;
;       X-37    RNG5037         Rod Gamache             25-Mar-1987
;               Fix test of spinlock index VS. FIPL in FORK_WAIT dispatcher.
;
;       X-36    SJF             Stu Farnham             20-Mar-1987
;        -35    Fix branch broken by longer FORKLOCK. Add $SPLCODDEF.
;
;       X-34    TLC5001         Thomas L. Cafarella     18-Mar-1987
;               Re-write disk queue length accumulation routine
;               (LOG_DISK_QLENS) so that it runs faster. This was
;               accomplished by eliminating the JSB to IOC$SCAN_IODB
;               and instead scanning the IO data base directly.
;
;       X-33    WCT0019         Ward C. Travis          17-Mar-1987
;               Add ENVIRON= parameter  to DSBINT and  SETIPL macro
;               uses to allow assembly-time  detection of neglected
;               uses.
;
;       X-32    SJF             Stu Farnham             17-Mar-1987
;               Remove assignment of new quantum on sanity timeout
;               of CPU in override set. Such CPUs increase their quantum
;               on entry to override, and set it back on exit. Also,
;               remove TPOINTER adjustments for CPUs which have left
;               the active set; those are now done as part of leaving.
;
;       X-31    MSH0303         Michael S. Harvey       26-Feb-1987
;               Rearrange, rewrite, and otherwise tweak the hardware
;               clock service routine to eke out greater performance.
;               Eliminate the interrupt stack busywait accounting. Fix
;               some bugs related to clock tick accounting and conditional
;               assembly statements. Revector the timer related SCB entries
;               to point directly to these service routines rather than to
;               a pointer to these service routines.
;
;       X-30    SJF             Stu Farnham             25-Feb-1987
;               Make CPUSANITY BUG_CHECK FATAL.
;
```

```
115 ;       X-29    SJF             Stu Farnham             12-Feb-1987
116 ;               Make SMP sanity timer aware to TIME_CONTROL sysgen
117 ;               parameter. Change SCH$PIXSCAN to SCH$ONE_SEC. Modify
118 ;               decrement of CPU$W_SANITY_TIMER to reflect revised
119 ;               semantics of parameters.
120 ;
121 ;       X-28    SF04002         Stephen Fiorelli        12-Feb-1987
122 ;               SCH$GL_NULLPCB becomes SCH$AR_NULLPCB.
123 ;
124 ;       X-27    RNG5027         Rod Gamache             3-Feb-1987
125 ;               Make FORK_WAIT queue a non-interlocked queue again.
126 ;
127 ;       X-26    SJF             Stu Farnham             3-Feb-1987
128 ;               Fix build breaker.
129 ;
130 ;       X-25    SJF             Stu Farnham             26-Jan-1987
131 ;               Add SMP CPU sanity timer logic.
132 ;
133 ;       X-24    HH243           Hai Huang               22-Jan-1987
134 ;               Set up EXE$GL_TQFL in initialization routine.
135 ;
136 ;       X-23    RNG5023         Rod N. Gamache          13-Jan-1987
137 ;               Make SCH$QEND run at IPL SCHED, rather than IPL 7.
138 ;               Remove SMP workaround code that allowed EXE$GQ_SYSTIME
139 ;               to be modified by a non-primary CPU. Since all SET TIME
140 ;               operations are now forced through the primary.
141 ;
142 ;       X-22    SSA0001         Stan Amway              12-Dec-1986
143 ;               Remove all PIXSCAN logic from this module. Invoke
144 ;               SCH$PIXSCAN to do the one second CPU scheduling logic.
145 ;
146 ;       X-21    RNG0021         Rod Gamache             3-Nov-1986
147 ;               Block SCHEDuling while setting the event flag and QUEUEING
148 ;               the ast for TIMER events.
149 ;
150 ;       X-20    HH0224          Hai Huang               29-Oct-1986
151 ;               Add time service support.
152 ;
153 ;       X-19    RNG0019         Rod Gamache             29-Oct-1986
154 ;               Change IPL$_SCHED to IPL$_RESCHED.
155 ;
156 ;       X-18    JWT0257         Jim Teague              17-Oct-1986
157 ;               Support for timer requests in CPU time.  Involved
158 ;               code that is required to fit the temporary SMP
159 ;               code at label REPTIM below.  When those two macros
160 ;               (LOCK and UNLOCK) are removed, the LOCK just
161 ;               below CHECK_CPU_TIME must also be removed.
162 ;
163 ;       X-17    RNG0017         Rod N. Gamache          8-Oct-1986
164 ;               Fix workaround bug and processing of TQEs on primary.
165 ;
166 ;       X-16    EMB             Ellen M. Batbouta       4-Oct-1986
167 ;               Fix broken branch.
168 ;
169 ;       X-15    SJF             Stu Farnham             2-Oct-1986
170 ;               Work around synchronization window with EXE$SETIME.
171 ;
```

```
                ;       X-12    RNG0012         Rod Gamache             15-Sep-1986
                ;               Make the timekeeper CPU only run the TQE expiration code.
                ;               This helps solve the problem of timeouts on devices with
                ;               affinity.
                ;
                ;       X-11    RNG0011         Rod Gamache             24-Jul-1986
                ;               Synchronize access to deadlock search queue and subsequent
                ;               subroutine calls.
                ;
                ;       X-10    EMB0243         Ellen M. Batbouta       22-Jul-1986
                ;               Conditionally release the device lock in the device
                ;               timeout code.
                ;
                ;       X-9     RNG0008         Rod Gamache             8-Jul-1986
                ;               Fix insidious problems generated by CMS MERGE.
                ;
                ;       X-8     SJF             Stu Farnham             30-Jun-1986
                ;               Resolve conflicts from merge of SMP into main line
                ;
                ;       X-6     WMC0002         Wayne Cardoza           18-Jun-1986
                ;
                ;       X-4D2   WMC0001         Wayne Cardoza           23-Jan-1986
                ;               Fix minor merge problems.
                ;
                ;       X-4D1   SF04001         Stephen Fiorelli        11-Dec-1985
                ;               Resolve conflicts from initial merge of exec reorg
                ;               thread and mainline (4.4 BL7).
                ;
                ;       X-1C4   TCM0001         Trudy C. Matthews       11-Oct-1985
                ;               Add initialization that places device timer queue entry
                ;               on timer queue.
                ;
                ;       X-4     TLC1095         Thomas L. Cafarella     04-Oct-1985     15:00
                ;               Do not accumulate disk queue length information
                ;               for a disk while it has a mount verify in progress.
                ;
                ;       X-3     TLC1094         Thomas L. Cafarella     16-Sep-1985     13:00
                ;               Eliminate shadow set member from set of disks for which
                ;               queue length information is collected.
                ;
                ;       X-2     TLC1093         Thomas L. Cafarella     16-Aug-1985     17:00
                ;               Add code to one-second timer routine to accurately maintain
                ;               disk queue length information.
                ;
                ;       X-1A24  WAL1012         William A. Laing        12-Feb-1986
                ;               Change CPU mode accounting so that the 6 modes
                ;               (KERN,INTSTK,EXEC,SUPR,USER,COMP) plus NULL all add up to
                ;               100 ticks per second. i.e. NULL time is now not part of
                ;               INTSTK time.
                ;
                ;       V03-013 CWH3012         CW Hobbs                29-Apr-1984
                ;               Change all W^SCH$Gxxx references to L^SCH$Gxxx to
                ;               fix current (and future) branch problems.
                ;
                ;       V03-012 WMC0012         Wayne Cardoza           23-Apr-1984
                ;               Declare pagedmeory and mailbox resources available once per
                ;               second just in case...
```

```
;
;       V03-011 CWH3011         CW Hobbs                14-Apr-1984
;               Fixed broken branch.
;
;       V03-010 SRB0118         Steve Beckhardt         26-Mar-1984
;               Fixed broken branches.
;
;       V03-009 SRB0117         Steve Beckhardt         17-Mar-1984
;               Removed loop around checking for locks on the timeout queue
;               needing a deadlock search.  This loop is now in the module
;               DEADLOCK.
;
;       V03-008 MIR0200         MICHAEL I. ROSENBLUM    15-OCT-1983
;               Remove the setipl to DIPL when RDUTIM expires
;               To allow the driver using this service to do it's
;               own synchronization.
;       V03-007 DWT0123         David W. Thiel          22-Aug-1983
;               Declare non-paged dynamic memory available once per
;               second.
;
;       V03-006 SRB0099         Steve Beckhardt         15-July-1983
;               Added loop to deadlock detection timeout code to allow
;               finding more than one deadlock per second.
;
;       V03-005 ROW0190         Ralph O. Weber           3-MAY-1983
;               Rewrite fork-and-wait processing using newly acquired
;               knowledge of how to move a queue from one header to another.
;               This better protects against infinite looping during the
;               processing of the fork-and-wait queue.
;
;       V03-004 ROW0176         Ralph O. Weber           4-APR-1983
;               Add code to process fork-and-wait queue to EXE$TIMEOUT.
;
```

```
                .SBTTL  DECLARATIONS
;
; MACRO LIBRARY CALLS
;

                $ACBDEF                         ;DEFINE ACB OFFSETS
                $CADEF                          ;DEFINE CONDITIONAL ASSEMBLY PARAMETERS
                $CPBDEF                         ;DEFINE CPU CAPABILITY CONSTANTS
                $CPUDEF                         ;DEFINE PER CPU OFFSETS
                $CRBDEF                         ;DEFINE CRB OFFSETS
                $DDBDEF                         ;DEFINE DDB OFFSETS
                $DEVDEF                         ;DEFINE DEVICE CHARACTERISTICS
                $DYNDEF                         ;DEFINE DYNAMIC STRUCTURE TYPES
                $ERLDEF                         ;DEFINE ERL OFFSETS
                $FKBDEF                         ;DEFINE FORK BLOCK OFFSETS
                $IPLDEF                         ;DEFINE INTERRUPT PRIORITY LEVELS
                $JIBDEF                         ;DEFINE JIB OFFSETS
                $LKBDEF                         ;DEFINE LKB OFFSETS
                $PCBDEF                         ;DEFINE PCB OFFSETS
                $PHDDEF                         ;DEFINE PHD OFFSETS
                $PRDEF                          ;DEFINE PROCESSOR REGISTERS
                $PR8SSDEF                       ;DEFINE SCORPIO PROCESSOR REGISTERS
                $PRIDEF                         ;DEFINE PRIORITY INCREMENTS
                $PSLDEF                         ;DEFINE PROCESSOR STATUS FIELDS
                $RSNDEF                         ;DEFINE RESOURCE WAIT NUMBERS
                $SBDEF                          ;DEFINE SB OFFSETS
                $SPLCODDEF                      ;DEFINE SPINLOCK CODES
                $STATEDEF                       ;DEFINE SCHEDULER STATE VALUES
                $TQEDEF                         ;DEFINE TQE OFFSETS
                $UCBDEF                         ;DEFINE UCB OFFSETS
                $VCBDEF                         ;DEFINE VCB OFFSETS
                $WQHDEF                         ; Wait queue header def'ns
                $TTYDEFS                        ; TTY UCB extension (must FOLLOW $UCBDEF)
                $TTYDEF                         ;DEFINE TTY SYMBOLS
```

```
TIMESCHDL - TIME DEPENDENT SCHEDULING 10-MAY-1989 16:56:06 VAX MACRO V5.0-8 Page 7
X-44U3 LOCAL DATA 1-NOV-1988 14:42:53 [SYS.SRC]TIMESCHDL.MAR;1 (3)

  298          .SBTTL  LOCAL DATA
  299
  300          DECLARE_PSECT   EXEC$NONPAGED_DATA
  301
  302          ASSUME  TQE$L_TQFL      EQ      0
  303          ASSUME  TQE$L_TQBL      EQ      4
  304          ASSUME  TQE$W_SIZE      EQ      8
  305          ASSUME  TQE$L_FPC       EQ      12
  306          ASSUME  TQE$L_FR3       EQ      16
  307          ASSUME  TQE$L_FR4       EQ      20
  308          ASSUME  TQE$Q_TIME      EQ      24
  309          ASSUME  TQE$Q_DELTA     EQ      32
  310
  311 ;
  312 ; PERMANENT TIME QUEUE ENTRY
  313 ;
  314
  315          .ALIGN  QUAD
  316 PERMENTRY:                                  ; PERMENENT TIME QUEUE ENTRY
  317          .ADDRESS EXE$GL_TQFL               ; FORWARD LINK TO LISTHEAD
  318          .ADDRESS EXE$GL_TQFL               ; BACK LINK TO DEVICE TIME OUT ENTRY
  319          .WORD   0                          ; SIZE OF ENTRY
  320          .BYTE   DYN$C_TQE                  ; TYPE OF DATA STRUCTURE
  321                                             ; This TQE cannot repeat.
  322          .BYTE   TQE$C_TMSNGL & <^CTQE$M_REPEAT> ; REQUEST TYPE OF ENTRY
  323          .BLKL   3                          ; THREE UNUSED LONGWORDS
  324          .LONG   ^X0FFFFFFFF                ; INFINITY EXPIRATION TIME
  325          .LONG   ^X0FFFFFFFF                ;
  326
  327 ;
  328 ; DEVICE TIME OUT TIME QUEUE ENTRY
  329 ;
  330
  331          .ALIGN  QUAD
  332 DEVICETIM:                                  ; DEVICE TIME OUT TIME QUEUE ENTRY
  333          .LONG   0                          ; FORWARD LINK
  334          .LONG   0                          ; BACK LINK
  335          .WORD   40                         ; SIZE OF ENTRY
  336          .BYTE   DYN$C_TQE                  ; TYPE OF DATA STRUCTURE
  337          .BYTE   TQE$C_SSREPT               ; REQUEST TYPE OF ENTRY
  338          .ADDRESS EXE$TIMEOUT               ; PC OF SYSTEM SUBROUTINE
  339          .ADDRESS IOC$GL_DEVLIST            ; ADDRESS OF I/O DATA BASE LISTHEAD
  340          .BLKL   1                          ; ONE UNUSED LONGWORD
  341          .QUAD   0                          ; QUAD WORD OF EXPIRATION TIME
  342          .LONG   100000*100                 ; DELTA REPEAT TIME OF 1 SECOND
  343          .LONG   0                          ;
  344
```

```
TIMESCHDL - TIME DEPENDENT SCHEDULING 10-MAY-1989 16:56:06 VAX MACRO V5.0-8 Page 8
X-44U3 INITIALIZATION ROUTINE 1-NOV-1988 14:42:53 [SYS.SRC]TIMESCHDL.MAR;1 (4)

                        .SBTTL  INITIALIZATION ROUTINE
                ;++
                ; INI$TIMER_QUEUE
                ;       This routine puts the permanent timer queue entry
                ;       and the device timer queue entry on the timer queue.
                ;
                ;       This routine revectors the timer pointers in the SCB
                ;       to point directly to the service routines in this loadable image.
                ;
                ;--
                        DECLARE_PSECT   EXEC$INIT_CODE
                INITIALIZATION_ROUTINE  INI$TIMER_QUEUE

                INI$TIMER_QUEUE::
                        MOVAB   G^EXE$GL_TQFL,R1        ; Addr of timer queue listhead.

                        MOVAB   PERMENTRY,R0            ; Address of permanent timer queue entry.
                        MOVL    R0,(R1)                 ; Make it the first entry in timer queue.
                        MOVL    R0,4(R1)                ; Set up 'EXE$GL_TQBL'
                        MOVL    R0,G^EXE$AR_TQENOREPT   ; Set up pointer to permanent timer queue en

                        MOVAB   DEVICETIM,R0            ; Address of device timer queue entry.
                        MOVQ    G^EXE$GQ_SYSTIME,-      ; Store systime in expiration field.
                                TQE$Q_TIME(R0)
                        INSQUE  TQE$L_TQFL(R0),-        ; Insert device timer entry immediately
                                TQE$L_TQFL(R1)          ; before permanent timer entry.

                ;
                ; Replace the timer vectors from the SCB that point to the base image
                ; with direct references to the interrupt service routines.
                ;
                        MOVL    G^EXE$GL_SCB,R0         ; Get address of the SCB

                        SUBL3   #1,^XC0(R0),R1          ; Get address of interval clock vector
                        CMPW    (R1)+,#^X<9F17>         ; Is this a JMP @# instruction?
                        BNEQ    10$                     ; If NEQ no, leave the SCB vector alone
                        ADDL3   #1,(R1),^XC0(R0)        ; Rewrite SCB vector with direct pointer

                10$:    SUBL3   #1,^X9C(R0),R1          ; Get address of software timer vector
                        CMPW    (R1)+,#^X<9F17>         ; Is this a JMP @# instruction?
                        BNEQ    20$                     ; If NEQ no, leave the SCB vector alone
                        ADDL3   #1,(R1),^X9C(R0)        ; Rewrite SCB vector with direct pointer

                20$:    MOVL    #SS$_NORMAL,R0          ; Return success status
                        RSB                             ;
```

```
  392          .SBTTL  MULTIPROCESSOR SANITY TIMER SERVICE
  393 ;+
  394 ;
  395 ; THIS ROUTINE IS CALLED FROM THE HARDWARE CLOCK SERVICE ROUTINE WHEN
  396 ; A CPU'S SANITY TIMER HAS EXPIRED.
  397 ;
  398 ;-
  399
  400          DECLARE_PSECT   EXEC$NONPAGED_CODE
  401
  402
  403          .ENABLE LSB
  404
  405 CHECK_SANITY_TIMER:
  406 ;
  407 ; The sanity timer of the CPU we are watching has expired. Check for
  408 ; sanity timeouts disabled, CPU removed from active set, CPU in override set, and
  409 ; recheck sanity timer before actually timing that CPU out.
  410 ;
  411          PUSHL   R2
  412          SUBL3   #CPU$W_SANITY_TIMER,-
  413                  CPU$L_TPOINTER(R1),R2   ; address of target CPU's database
  414          BBS     #EXE$V_NOSMPSANITY,-
  415                  g^EXE$GL_TIME_CONTROL,1$ ; new timeout quantum if timeouts disabled
  416
  417          ADAWI   #0,CPU$W_SANITY_TIMER(R2) ; check one last time
  418          BGTR    2$
  419          JSB     G^SMP$CONTROLP_CPUS     ; any CPUs halted by operator?
  420          TSTL    R0
  421          BNEQ    1$                      ; Yes, assign new quantum.
  422          BUG_CHECK CPUSANITY,FATAL
  423
  424 ;
  425 ; Assign a new sanity quantum.
  426 ;
  427
  428 1$:      SUBW3   CPU$W_SANITY_TIMER(R2),-
  429                  G^SGN$GW_SMP_SANITY_CNT,-
  430                  R0                      ; calculate # cycles expired
  431          ADAWI   R0,CPU$W_SANITY_TIMER(R2) ; add back into timer
  432 2$:      MOVL    (SP)+,R2                ; restore register
  433          BRB     10$                     ; continue
  434
  435 3$:      BRB     CHECK_SANITY_TIMER
```

```
437          .SBTTL  HARDWARE CLOCK INTERRUPTS
438 ;+
439 ; EXE$HWCLKINT - HARDWARE CLOCK INTERRUPT
440 ;
441 ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN THE CLOCK COUNT REGISTER
442 ; OVERFLOWS. THE CURRENT ABSOLUTE TIME IS UPDATED, THE SMP SANITY TIMER
443 ; IS CHECKED, THE ACCOUNTING INTEGRAL OF THE CURRENT PROCESS IS INCREMENTED,
444 ; AND THE QUANTUM OF THE CURRENT PROCESS IS INCREMENTED. IF THE PROCESS
445 ; QUANTUM TRANSISTS TO ZERO OR THE FIRST ENTRY IN THE TIMER QUEUE CAN BE
446 ; REMOVED, THEN A SOFTWARE INTERRUPT IS REQUESTED ON THE TIMER LEVEL.
447 ; THE CLOCK INTERRUPT IS THEN DISMISSED.
448 ;
449 ;       THE CLOCK IS CURRENTLY SET TO INTERRUPT AT 10MS INTERVALS.
450 ;-
451          .ALIGN  LONG
452
453          UNIVERSAL_SYMBOL        EXE$HWCLKINT
454 ;EXE$HWCLKINT::                              ;HARDWARE CLOCK INTERRUPT
455          MTPR    #^X800000C1,#PR$_ICCS       ;CLEAR INTERRUPT + ERROR AND RE-ENABLE
456          NOP                                 ;FORCE NEXT INST. TO LONGWORD ALIGN
457
458          UNIVERSAL_SYMBOL        EXE$UBCLKINT
459 ;EXE$UBCLKINT::
460          MOVQ    R0,-(SP)                    ;Save R0,R1
461          FIND_CPU_DATA   R1,-                ;Get this CPU's per-CPU database address
462                          ISTACK=YES          ; (Let macro assume we're on IS)
463 ;
464 ; If this is a multiprocessor configuration, execute the operation of this
465 ; CPU's part of the sanity timer mechanism. This may or may not be the
466 ; PRIMARY (timekeeper) CPU.
467 ;
468 ; If this is not a multiprocessor configuration, skip the sanity timer code
469 ; and execute timer functions that are reserved to the PRIMARY CPU in a
470 ; multiprocessor configuration.
471 ;
472          ASSUME SMP$V_ENABLED EQ 0
473          BLBC    G^SMP$GL_FLAGS,20$          ;If LBC, skip sanity timer mechanisms
474 ;
475 ; Reset this CPU's sanity timer.
476 ;
477          SUBW3   CPU$W_SANITY_TIMER(R1),-
478                  G^SGN$GW_SMP_SANITY_CNT,-
479                  R0                          ; calculate # cycles expired
480          ADAWI   R0,CPU$W_SANITY_TIMER(R1) ;and add back in to timer
481 ;
482 ; Count down number of ticks until next sanity timer check
483 ;
484          DECW    CPU$W_SANITY_TICKS(R1)  ; count towards 0
485          BGTR    10$                     ; If GTR, not expired
486 ;
487 ; Reset ticks until next sanity event. Check the sanity timer of the CPU
488 ; we are watching for possible timeout.
489 ;
490          MOVW    G^SGN$GW_SMP_TICK_CNT,-
491                  CPU$W_SANITY_TICKS(R1)  ; reset tick counter
492          MNEGW   G^SGN$GW_SMP_TICK_CNT,- ; calculate decrement amount
493                  R0
```

```
  494          ADAWI   R0,@CPU$L_TPOINTER(R1)  ; count down other CPU's timer
  495          BLEQ    3$                      ; If LEQ, expired. Check special conditions
  496
  497 ;
  498 ; Carry out system time related work. Only the PRIMARY (timekeeper) CPU in a
  499 ; multiprocessor configuration does this work, and only in the multiprocessor
  500 ; case must a spinlock be used for synchronization.
  501 ;
  502 ; The commented code below is replaced by the optimal portions that are
  503 ; required in this environment, for the sake of performance.
  504 ;
  505 ;        LOCK    LOCKNAME=HWCLK,-        ;Lock hardware clock data structures
  506 ;                CONDITION=NOSETIPL,-    ;Don't set IPL we are already at HWCLK
  507 ;                PRESERVE=NO             ;Don't preserve R0
  508 ;
  509
  510          ASSUME  CPB$C_PRIMARY EQ 0
  511 10$:     BLBC    CPU$L_CAPABILITY(R1),40$ ;If LBC, this is a non-PRIMARY CPU
  512          MOVZBL  #SPL$C_HWCLK,R0         ;Load hardware clock spinlock index
  513          JSB     G^SMP$ACQUIRE           ;Lock the HWCLK related data structures
  514
  515 ;
  516 ; Execute system timer related functions. This is either the PRIMARY CPU in
  517 ; a multiprocessor configuration, or the only CPU in a uniprocessor.
  518 ;
  519 20$:     ADDL2   G^EXE$GL_TICKLENGTH,-
  520                  G^EXE$GQ_SYSTIME        ;Update system absolute time
  521          ADWC    #0,G^EXE$GQ_SYSTIME+4   ;
  522          INCL    G^EXE$GL_ABSTIM_TICS    ;Maintain scheduler's high res clock
  523          TSTL    G^EXE$GL_TIMEADJUST     ;Count down adjustment
  524          BEQL    25$                     ;If eql, no adjustment necessary
  525          SOBGTR  G^EXE$GL_TIMEADJUST,25$ ;Branch if no adjustment necessary
  526          MOVL    G^EXE$GL_SYSTICK,G^EXE$GL_TICKLENGTH ;Restore standard tick
  527
  528 25$:     CMPL    G^EXE$GQ_1ST_TIME+4,-
  529                  G^EXE$GQ_SYSTIME+4      ;COMPARE HIGH ORDER PARTS OF TIME
  530          BGTRU   35$                     ;IF GTRU ENTRY NOT DUE
  531          BLSSU   30$                     ;IF LSSU ENTRY DUE
  532          CMPL    G^EXE$GQ_1ST_TIME,-
  533                  G^EXE$GQ_SYSTIME        ;COMPARE LOW ORDER PARTS OF TIME
  534          BGTRU   35$                     ;IF GTRU ENTRY NOT DUE
  535 30$:     SOFTINT #IPL$_TIMERFORK         ;REQUEST SOFTWARE INTERRUPT ON TIMER LEVEL
  536 35$:                                     ;
  537 ;
  538 ; If this is a uniprocessor configuration, then we got here without taking
  539 ; out the HWCLK spinlock, so we don't want to release it either.
  540 ;
  541 ; The commented code below is replaced by the optimal portions that are
  542 ; required in this environment, for the sake of performance.
  543 ;
  544 ;        UNLOCK  LOCKNAME=HWCLK,-        ;Release the hardware clock lock
  545 ;                PRESERVE=NO             ; if this is a multiprocessor system
  546 ;
  547          ASSUME  SMP$V_ENABLED EQ 0
  548          BLBC    G^SMP$GL_FLAGS,40$      ;If LBC, no SMP LOCK/UNLOCK required
  549          MOVZBL  #SPL$C_HWCLK,R0         ;Load hardware clock spinlock index
  550          JSB     G^SMP$RELEASE           ;Unlock the HWCLK spinlock
```

```
551
552 ;
553 ; If measurement is enabled, update timer statistics fields. All processors
554 ; maintain their own set of time measurement statistics.
555 ;
556 40$:
557         .IF NE  CA$_MEASURE
558
559         CVTBL   15(SP),R0               ;Get upper byte of saved PSL
560         BGEQ    45$                     ;If GEQ, CM bit not set
561         MOVZBL  #5,R0                   ;Else insert CM index
562 45$:   BICB2   #^XF8,R0                ;Convert extended byte to index
563         TSTB    CPU$B_BUSYWAIT(R1)      ;In busywait at any IPL?
564         BEQL    50$                     ;If EQL no
565         BBS     #PSL$V_IS,12(SP),47$    ;If busywait on interrupt stack, branch
566 46$:   INCL    CPU$L_MPSYNCH(R1)       ;Count a true busywait tick
567         MOVQ    (SP)+,R0                ;Restore R0,R1
568         REI                             ;Dismiss tick
569
570 47$:   CMPZV   #PSL$V_IPL,#PSL$S_IPL,- ;Check if at IPL SCHED or lower.
571                 12(SP),#IPL$_SCHED      ;
572         BGTR    46$                     ;GTR than IPL SCHED, count as busywait
573         BBC     #CPU$V_SCHED,CPU$B_FLAGS(R1),46$
574                                         ;True busywait if scheduling flag not set
575         INCL    CPU$L_NULLCPU(R1)       ;Accumulate NULL CPU time this processor
576         INCL    CPU$L_KERNEL(R1)[R0]    ;Increment per-CPU statistics vector
577         MOVQ    (SP)+,R0                ;Restore R0,R1
578         REI                             ;Dismiss tick
579
580 50$:   INCL    CPU$L_KERNEL(R1)[R0]    ;Increment per-CPU statistics vector
581
582         .IFF
583
584         TSTB    CPU$B_BUSYWAIT(R1)      ;Is this an overhead tick?
585         BNEQ    60$                     ;If NEQ yes, don't add to process quantum
586         .ENDC
587
588         MOVL    CPU$L_CURPCB(R1),R0     ;Get current process PCB
589         CMPL    R0,G^SCH$AR_NULLPCB     ;Is this the NULL job?
590         BEQL    70$                     ;If EQL yes, go handle NULL case
591         BBS     #PSL$V_IS,12(SP),60$    ;If set, count this as overhead tick
592         MOVL    PCB$L_PHD(R0),R0        ;Get address of process header
593         INCL    PHD$L_CPUTIM(R0)        ;Increment accounting integral
594         INCW    PHD$W_QUANT(R0)         ;Increment time quantum
595         BLSS    60$                     ;If LSS, quantum still good
596         SOFTINT #IPL$_TIMERFORK         ;Request software interrupt on timer level
597 60$:
598
599         .IF NE  CA$_MEASURE
600
601         MOVQ    (SP)+,R0                ;Restore R0,R1
602         REI                             ;Dismiss this interrupt
603
604 70$:   CMPZV   #PSL$V_IPL,#PSL$S_IPL,-
605                 12(SP),#IPL$_RESCHED    ;Was NULL job really interrupted?
606         BNEQ    60$                     ;If NEQ no, count as overhead tick
607         INCL    CPU$L_NULLCPU(R1)       ;Accumulate NULL CPU time this processor
```

```
  608          MOVQ    (SP)+,R0                ;Restore previously saved registers
  609          REI                             ;Dismiss this interrupt
  610
  611          .IFF
  612
  613 70$:     MOVQ    (SP)+,R0                ;Restore previously saved registers
  614          REI                             ;Dismiss this interrupt
  615
  616          .ENDC
  617
  618
  619          .DISABLE        LSB
```

```
                .SBTTL  SOFTWARE TIMER INTERRUPTS
;+
; EXE$SWTIMINT - SOFTWARE TIMER INTERRUPTS
;
; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A SOFTWARE INTERRUPT IS
; REQUESTED ON THE TIMER LEVEL. TIMER INTERRUPTS ARE REQUESTED WHEN THE
; CURRENT PROCESS HAS EXCEEDED ITS CPU TIME QUANTUM OR THE FIRST ENTRY IN
; THE TIMER QUEUE IS DUE.
;-

        .ENABL  LSB
        .ALIGN  LONG
        UNIVERSAL_SYMBOL        EXE$SWTIMINT
;EXE$SWTIMINT::                          ;SOFTWARE TIMER INTERRUPTS
        PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ;SAVE REGISTERS R0 THRU R5
        FIND_CPU_DATA   R1,-            ;Get base of per CPU data structure
                        ISTACK=YES      ; (Let macro assume we're on IS)

        MOVL    CPU$L_CURPCB(R1),R4     ;GET THIS CPU's CURRENT PROCESS PCB ADDRESS
        MOVL    PCB$L_PHD(R4),R5        ;GET ADDRESS OF PROCESS HEADER
        TSTW    PHD$W_QUANT(R5)         ;QUANTUM END?
        BLSS    5$                      ;IF LSS NO
        LOCK    LOCKNAME=SCHED,-        ;Acquire Scheduler lock
                PRESERVE=NO             ;Don't preserve R0
        PUSHL   R1                      ;Save CPU data area address
        JSB     G^SCH$QEND              ;CALL SCHEDULER TO RESET QUANTUM
        POPL    R1                      ;Restore CPU data area address
        UNLOCK  LOCKNAME=SCHED,-        ;Release Scheduler lock
                PRESERVE=NO             ;Don't preserve R0

;
; Only process TQE expirations on the PRIMARY CPU. This is
; to allow device affinity to work - with the PRIMARY CPU
; handling device timeouts.
;

        ASSUME  CPB$C_PRIMARY EQ 0
5$:     BLBC    CPU$L_CAPABILITY(R1),15$ ;If LBC, this is a non-PRIMARY CPU

CHKTMQ: LOCK    LOCKNAME=TIMER,-        ;Acquire timer lock
                PRESERVE=NO             ;Don't preserve R0
        LOCK    LOCKNAME=HWCLK,-        ;Lock hardware clock data structures
                PRESERVE=NO             ;Don't preserve R0
        CMPL    G^EXE$GQ_1ST_TIME+4,-
                G^EXE$GQ_SYSTIME+4      ;COMPARE HIGH ORDER PARTS OF TIME
        BLSSU   20$                     ;IF LSSU ENTRY IS DUE
        BGTRU   10$                     ;IF GTRU ENTRY IS NOT DUE
        CMPL    G^EXE$GQ_1ST_TIME,-
                G^EXE$GQ_SYSTIME        ;COMPARE LOW ORDER PARTS OF TIME
        BLEQU   20$                     ;IF LEQU ENTRY IS DUE
10$:    UNLOCK  LOCKNAME=HWCLK,-        ;unlock HWCLK data structure and
                NEWIPL=#IPL$_TIMER,-    ;lower IPL to TIMER
                PRESERVE=NO             ;Don't preserve R0
        UNLOCK  LOCKNAME=TIMER,-        ;Unlock TIMER spinlock
                PRESERVE=NO             ;Don't preserve R0
15$:    POPR    #^M<R0,R1,R2,R3,R4,R5>  ;RESTORE REGISTERS R0 THRU R5
        REIMAC                          ;
```

```
678
679 ;
680 ; REMOVE DUE ENTRY FROM TIMER QUEUE
681 ;
682 ;       There is an assumption here that if a TQE entry expires then there is at
683 ;       least 1 more entry in the queue that never expires. This should always
684 ;       be true. There is a permanent timer entry that never expires PERMENTRY.
685
686 20$:    MOVAL   G^EXE$GL_TQFL,R0        ;GET ADDRESS OF TIME QUEUE HEADER
687         REMQUE  @(R0),R5                ;REMOVE FIRST ENTRY FROM TIME QUEUE
688         MOVL    (R0),R0                 ;Get address of new head of timer queue
689         MOVQ    TQE$Q_TIME(R0),-        ;Copy time from new head of
690                 G^EXE$GQ_1ST_TIME       ;timer queue to first time
691         UNLOCK  LOCKNAME=HWCLK,-        ;unlock HWCLK data structure and
692                   NEWIPL=#IPL$_TIMER,-  ;lower IPL to TIMER
693                   PRESERVE=NO           ;Don't preserve R0
694         UNLOCK  LOCKNAME=TIMER,-        ;Unlock timer lock
695                   PRESERVE=NO           ;Don't preserve R0
696 ;
697 ;SMP NOTE that in an SMP system no locks are held at this point but we are at
698 ;     IPL$_TIMER which equals IPL$_SYNCH.
699 ;
700         ASSUME  IPL$_TIMER  EQ  IPL$_SYNCH
701         EXTZV   #0,#2,TQE$B_RQTYPE(R5),R0 ;GET REQUEST TYPE
702         CASE    R0,<TIMER,SYSUB,WAKEUP> ;DISPATCH TO PROCESSING ROUTINE
703         BUG_CHECK INVTQEFMT             ;INVALID TIME QUEUE ENTRY FORMAT
704
705 CHKTMQ_BR:                              ; LONG BRANCH TO CHKTMQ
706         BRW     CHKTMQ                  ;
707
708         .DSABL  LSB
709
710

711 ;
712 ; PROCESS SYSTEM SUBROUTINE
713 ;
714 ; **** WARNING ****
715 ;
716 ; Upon return from the system subroutine call, this routine expects R5 to
717 ; contain the address of a valid timer queue entery.  The TQE$V_REPEAT bit
718 ; of that TQE will be tested, and if it is set, the TQE will be reentered in
719 ; the timer queue.  Therefore, the call system subroutine CANNOT use the TQE,
720 ; pointed to by R5 at entry, for some other purpose and return here without
721 ; placing the address of a valid TQE in R5.
722 ;
723 ; To this end, the executive system table contains a global symbol,
724 ; EXE$AR_TQENOREPT, which contains the address of a always valid always
725 ; non-repeating timer queue entry.  Timer system subroutines wishing to use
726 ; the TQE which caused them to be called for purposes other than continued
727 ; use of the timer queue may load the address contained in EXE$AR_TQENOREPT
728 ; into R5 before returing to this routine.  This prevents duplicate use of
729 ; the TQE block which resulted in the system subroutine being called.
730 ;
731 ; For example, a system subroutine which decides to discontinue its timed
732 ; operations and deallocate the TQE would execute at least the following
733 ; instructions:
734 ;       ...
```

```
735 ;       MOVL    R5, R0                  ;NB: this uses the TQE
736 ;       JSB     G^COM$DRVDEALMEM        ;    as an IPL 6 fork block.
737 ;       ...
738 ;       MOVL    G^EXE$AR_TQENOREPT, R5  ;Setup no repeat TQE.
739 ;       RSB
740
741 ;
742 ;SMP NOTE the SYSTEM SUBROUTINE must acquire its own SMP locks
743 ;
744 SYSUB:  MOVQ    TQE$L_FR3(R5),R3        ;LOAD SUBROUTINE CONTEXT
745         JSB     @TQE$L_FPC(R5)          ;CALL SYSTEM SUBROUTINE
746         JSB     G^SMP$NOLOCKS           ;Verify no Spinlocks still held
747         BBC     #TQE$V_REPEAT,TQE$B_RQTYPE(R5),CHKTMQ_BR ;IF CLR, NOT REPEATABLE
748         BRW     REPTIM                  ;INSERT REPEAT REQUEST IN TIME QUEUE
749
750 ;
751 ; PROCESS TIMER
752 ;
753
754         .ENABL  LSB
755
756 CHECK_CPU_TIME:
757 ;SMP NOTE  PCBVEC and PCB being accessed here
758         MOVZWL  R1,R0                   ; Get process index
759         MOVL    @L^SCH$GL_PCBVEC[R0],R4 ; Convert PID to PCB addr
760         CMPL    R1,PCB$L_PID(R4)        ; Do PIDs match?
761         BEQL    1$
762         BRW     30$                     ; BR if process not in system
763 1$:     BBC     #PCB$V_RES,-
764                 PCB$L_STS(R4),2$        ; BR if process not resident
765         MOVL    PCB$L_PHD(R4),R2        ; Get PHD address
766         MOVL    PHD$L_CPUTIM(R2),R2     ; Get CPU time from PHD
767         BRB     3$
768 2$:     MOVL    PCB$L_CPUTIM(R4),R2     ; Use CPU time as of outswap
769 3$:     SUBL3   R2,TQE$L_CPUTIM(R5),R0  ; Compute min time remaining until due
770         BLEQ    5$                      ; BR if due *now*
771                                         ; R0 contains delta time in 10-ms units
772         EMUL    R0,#<10*1000*10>,#0,R0  ; Cvt delta to 100-ns units in R0,R1
773         BRW     35$                     ; Join common code to insert TQE
774
775 ;
776 ;SMP NOTE  a JIB is referenced, the PCBVEC is referenced, a PCB is referenced,
777 ;            the TQE is possibly deallocated
778 ;
779 TIMER:  MOVL    TQE$L_PID(R5),R1        ;GET TARGET PROCESS ID
780
781         BBS     #TQE$V_CHK_CPUTIM,-     ; Was this entry in terms of
782                 TQE$B_RQTYPE(R5),-      ;   CPU time?
783                 CHECK_CPU_TIME          ; If so, go check it
784
785 5$:     MOVZBL  #PRI$_TIMER,R2          ;SET PRIORITY INCREMENT CLASS
786         MOVZBL  TQE$B_EFN(R5),R3        ;GET EVENT FLAG NUMBER
787         ASSUME  IPL$_TIMER  EQ  IPL$_SCHED
788         LOCK    LOCKNAME=SCHED,-        ;BLOCK THE SCHEDULER
789                 CONDITION=NOSETIPL,-    ;DON'T BOTHER WITH RAISING IPL
790                 PRESERVE=NO             ;DON'T PRESERVE R0
791         JSB     G^SCH$POSTEF            ;POST EVENT FLAG
```

```
  792          BLBC    R0,30$                  ;IF LBC PROCESS NO LONGER IN SYSTEM
  793 ;SMP NOTE JIB being accessed here
  794          MOVL    PCB$L_JIB(R4),R0        ;GET JIB ADDRESS
  795
  796          ADAWI   #1,JIB$W_TQCNT(R0)      ; Increment quota
  797
  798          BBCCI   #1,JIB$B_FLAGS(R0),8$   ; If anyone else is waiting
  799          BSBW    RELEASE_WAITERS         ;  on TQCNT, start them up...
  800
  801 8$:      BBC     #ACB$V_QUOTA,TQE$B_RMOD(R5),20$ ;IF CLR, NO AST SPECIFIED
  802          MOVB    TQE$B_RMOD(R5),TQE$B_RQTYPE(R5) ;SET AST ACCESS MODE
  803          MOVZBL  #PRI$_TIMER,R2          ;SET PRIORITY INCREMENT CLASS
  804          JSB     G^SCH$QAST              ;QUEUE AST FOR PROCESS
  805          UNLOCK  LOCKNAME=SCHED,-        ;UNLOCK SCHEDULING DATABASE
  806                  PRESERVE=NO             ;DON'T PRESERVE R0
  807          BRW     CHKTMQ                  ;
  808
  809 10$:     ; SCHED spinlock held when entered here
  810
  811          MOVZWL  TQE$L_RQPID(R5),R4      ;GET REQUESTING PROCESS INDEX
  812          MOVL    @W^SCH$GL_PCBVEC[R4],R4 ;GET PROCESS PCB ADDRESS
  813          CMPL    PCB$L_PID(R4),TQE$L_RQPID(R5) ;PROCESS ID MATCH?
  814          BNEQ    30$                     ;IF NEQ NO
  815          ADAWI   #1,PCB$W_ASTCNT(R4)     ;UPDATE AVAILABLE AST QUEUE ENTRIES
  816          BRB     30$                     ;
  817 20$:     MOVZBL  #RSN$_ASTWAIT,R0        ;SET AST WAIT RESOURCE NUMBER
  818          JSB     G^SCH$RAVAIL            ;DECLARE RESOURCE AVAILABLE
  819
  820 30$:     ; SCHED spinlock held when entered here
  821
  822          UNLOCK  LOCKNAME=SCHED,-        ;UNLOCK SCHEDULING DATABASE
  823                  PRESERVE=NO             ;DON'T PRESERVE R0
  824          MOVL    R5,R0                   ;SET ADDRESS OF BLOCK TO DEALLOCATE
  825          JSB     G^EXE$DEANONPAGED       ;DEALLOCATE TIME QUEUE ENTRY
  826          BRW     CHKTMQ                  ;
  827
  828 ;
  829 ; PROCESS WAKE UP
  830 ;
  831
  832 WAKEUP: MOVL     TQE$L_PID(R5),R1        ;GET TARGET PROCESS ID
  833          ASSUME  IPL$_TIMER  EQ  IPL$_SCHED
  834          LOCK    LOCKNAME=SCHED,-        ;Acquire Scheduler lock
  835                  CONDITION=NOSETIPL,-
  836                  PRESERVE=NO             ;Don't preserve R0
  837          JSB     G^SCH$WAKE              ;WAKE PROCESS
  838          BLBC    R0,10$                  ;IF LBC PROCESS NOT IN SYSTEM
  839          BBC     #TQE$V_REPEAT,TQE$B_RQTYPE(R5),10$ ;IF CLR, THEN NOT REPEATABLE
  840          UNLOCK  LOCKNAME=SCHED,-        ;Release Scheduler lock
  841                  PRESERVE=NO             ;Don't preserve R0
  842 REPTIM:
  843          ;
  844          ; There is an interesting problem caused by the following code
  845          ; path. If EXE$GQ_SYSTIME can be modified while a TQE is not
  846          ; on the TQE list, then the TQE$Q_TIME field will not get updated
  847          ; and the TQE will mis-fire. The only known case of this occurance
  848          ; is during the power-up sequence after a powerfailure, if the
```

```
849          ; powerfailure occurred while the (a) CPU was in this code
850          ; path.
851          ;
852          MOVQ    TQE$Q_DELTA(R5),R0      ;GET DELTA REPEAT TIME
853 35$:     ADDL    TQE$Q_TIME(R5),R0       ;ADD LOW ORDER PARTS OF TIME
854          ADWC    TQE$Q_TIME+4(R5),R1     ;ADD HIGH ORDER PARTS OF TIME
855          JSB     G^EXE$INSTIMQ           ;INSERT ENTRY IN TIME QUEUE
856 40$:     BRW     CHKTMQ                  ;
857
858 RELEASE_WAITERS:
859 ;
860 ; This subroutine checks the MWAIT queue for other processes waiting
861 ;       on resources in this JIB.
862 ;
863 ; SCHED is owned for the duration of this subroutine
864
865          PUSHR   #^M<R0,R1,R2,R3,R4>     ; Save regs
866          MOVAL   G^SCH$GQ_MWAIT,R3       ; Address of MWAIT Q
867          MOVL    (R3),R4                 ; Head PCB
868 100$:    MOVZBL  #PRI$_RESAVL,R2         ; Priority increment
869          CMPL    R3,R4                   ; End of queue?
870          BEQL    300$                    ; Branch if end
871          CMPL    (SP),PCB$L_EFWM(R4)     ; Is process waiting on this JIB?
872          BNEQ    200$                    ; No, skip it
873          PUSHL   (R4)                    ; Save forward link
874          JSB     G^SCH$CHSE              ; Change state to executable
875          DECW    WQH$W_WQCNT(R3)         ; 1 less entry in queue
876          POPL    R4                      ; Restore forward link
877          BRB     100$                    ;  and continue
878 200$:    MOVL    (R4),R4                 ; Forward link to next PCB
879          BRB     100$                    ;  and continue
880 300$:    POPR    #^M<R0,R1,R2,R3,R4>     ; Restore regs
881          RSB                             ; Return
882
883          .DSABL  LSB
```

```
  885          .SBTTL  SEARCH FOR TIME OUTS
  886 ;+
  887 ; EXE$TIMEOUT - SEARCH FOR TIME OUTS
  888 ;
  889 ; THIS ROUTINE IS ENTERED ONCE A SECOND TO PERFORM VARIOUS FUNCTIONS THAT
  890 ; NEED TO BE PERFORMED ONCE A SECOND.  THESE INCLUDE:
  891 ;
  892 ;       1)  SCAN THE DEVICE DATABASE FOR DEVICES THAT MAY HAVE TIMED OUT
  893 ;       2)  SCAN FOR CRB'S  THAT MAY HAVE TIMED OUT
  894 ;       3)  SCAN FOR WAITING LOCKS THAT MAY HAVE TIMED OUT (INITIATE
  895 ;           DEADLOCK SEARCH)
  896 ;       4)  WAKE THE SWAPPER, IF NECESSARY
  897 ;       5)  WAKE THE ERROR LOG PROCESS, IF NECESSARY
  898 ;       6)  SCAN FOR MEMORY CRD ERRORS AND REENABLE MEMORY INTERRUPTS
  899 ;       7)  UPDATE THE SYSTEM ABSOLUTE TIME IN SECONDS
  900 ;       8)  DECLARE A NON-PAGED DYNAMIC MEMORY AVAILABLE EVENT
  901 ;-
  902
  903          UNIVERSAL_SYMBOL        EXE$TIMEOUT
  904 ;EXE$TIMEOUT::                           ;SEARCH FOR TIME OUTS
  905 ;
  906 ;SMP NOTE the DDBs are searched, the UCBs may be searched, SCH$SWPWAKE is
  907 ;            called, a bit in ERL$GB_BUFFLAG is checked uninterlocked and
  908 ;            ERL$WAKE is optionally called, ECC$REENABLE is called
  909
  910          PUSHL   R6                      ;SAVE A REGISTER
  911          PUSHL   R5                      ;SAVE ANOTHER
  912
  913          ; Do the next instruction here, so that R3 can be destroyed!
  914          MOVL    DDB$L_LINK(R3),R6       ;GET ADDRESS OF FIRST DDB
  915
  916          ASSUME  IPL$_TIMER  EQ  IPL$_SCHED
  917          LOCK    LOCKNAME=SCHED,-        ;Acquire Scheduler lock
  918                  CONDITION=NOSETIPL,-
  919                  PRESERVE=NO             ;Don't preserve R0
  920          JSB     G^SCH$SWPWAKE           ;WAKE SWAPPER IF NECESSARY
  921          UNLOCK  LOCKNAME=SCHED,-        ;Release Scheduler lock
  922                  PRESERVE=NO             ;Don't preserve R0
  923          INCL    G^EXE$GL_ABSTIM         ;UPDATE ABSOLUTE TIME IN SECONDS
  924          BBC     #ERL$V_TIMER,G^ERL$GB_BUFFLAG,10$ ;IF CLR, TIMER NOT ACTIVE
  925          JSB     G^ERL$WAKE              ;WAKE ERROR LOG FORMAT PROCESS
  926 10$:     JSB     G^ECC$REENABLE          ;WAKE CRD INTERRUPT REENABLE AND SCAN
  927
  928 ;
  929 ; SCAN FOR DEVICE TIMEOUTS
  930 ;
  931 20$:     MOVL    DDB$L_UCB(R6),R5        ;GET ADDRESS OF FIRST UCB
  932          BEQL    60$                     ;EQL MEANS NO UCB'S AS YET ON THIS DDB
  933          BBS     #DEV$V_MBX,UCB$L_DEVCHAR(R5),60$ ;IF SET, DEVICE IS MAILBOX
  934 30$:     BBS     #UCB$V_TIM,UCB$W_STS(R5),70$ ;IF SET, TIME OUT ENABLED
  935 40$:     BBC     #DEV$V_TRM,UCB$L_DEVCHAR(R5),50$ ;IF CLR, DEVICE NOT TERMINAL
  936          BBS     #UCB$V_TT_TIMO,UCB$W_DEVSTS(R5),67$ ;IF SET, READ TIMEOUT ENABLED
  937 50$:     MOVL    UCB$L_LINK(R5),R5       ;GET ADDRESS OF NEXT UCB
  938          BNEQ    30$                     ;IF NEQ MORE TO SCAN
  939 60$:     MOVL    DDB$L_LINK(R6),R6       ;GET ADDRESS OF NEXT DDB
  940          BNEQ    20$                     ;IF NEQ MORE TO SCAN
  941
```

```
942 ;
943 ; FINISHED DEVICE SCAN, NOW CHECK THE LIST OF CRB'S
944 ;
945
946          MOVAL   G^IOC$GL_CRBTMOUT,R6    ;PICK UP LIST HEAD
947 65$:     MOVL    (R6),R6                 ;ANY MORE TO SCAN?
948          BEQL    68$                     ;NO, DONE
949          FORKLOCK LOCK=CRB$B_FLCK-CRB$L_TIMELINK(R6) ;Lock FORK threads/use IPL from
950          CMPL    CRB$L_DUETIME-CRB$L_TIMELINK(R6),-
951                  G^EXE$GL_ABSTIM         ;YES, IS THIS ONE DUE?
952          BGTRU   66$                     ;NO, SCAN AGAIN
953          MOVAL   -CRB$L_TIMELINK(R6),R3  ;YES, PICK UP POINTER TO CRB
954          MNEGL   #1, CRB$L_DUETIME(R3)   ;SET FOR NO MORE TIMEOUTS
955          JSB     @CRB$L_TOUTROUT(R3)     ;CALL THE TIMEOUT ROUTINE
956 66$:     FORKUNLOCK LOCK=CRB$B_FLCK-CRB$L_TIMELINK(R6),- ;Unlock FORK threads
957                  NEWIPL=#IPL$_TIMER      ;Reset the IPL
958          BRB     65$                     ;CONTINUE SCAN
959 ;
960 67$:     BRW     90$                     ;BRANCH HELPER
961
962 68$:     BRW     LOG_DISK_QLENS          ;BRANCH HELPER
963
964 69$:     BRW     40$                     ;BRANCH ASSIST
965 ;
966 ; DEVICE HAS ENABLED TIME OUT - SEE IF IT HAS TIMED OUT
967 ;
968
969 70$:     CMPL    UCB$L_DUETIM(R5),G^EXE$GL_ABSTIM ;POSSIBLE TIME OUT?
970          BGTRU   69$                     ;IF GTRU NO
971          ASSUME  UCB$B_FIPL  EQ  UCB$B_FLCK
972          ASSUME  SPL$_MIN_INDEX  EQ  32
973          BBC     #5,UCB$B_FLCK(R5),75$   ;Br if not a forklock index
974          FORKLOCK LOCK=UCB$B_FLCK(R5)    ;Lock FORK threads/use IPL from FLCK
975 75$:
976          DEVICELOCK LOCKADDR=UCB$L_DLCK(R5) ; Lock interrupt threads/use IPL
977          SETIPL  #IPL$_POWER,-           ;Raise IPL to POWER
978                  ENVIRON=UNIPROCESSOR
979          BBC     #UCB$V_TIM,UCB$W_STS(R5),80$ ;IF CLR, THEN TIME OUT NOT ENABLED
980          CMPL    UCB$L_DUETIM(R5),G^EXE$GL_ABSTIM ;DEVICE TIME OUT?
981          BGTRU   80$                     ;IF GTRU NO
982          BICW    #UCB$M_INT!UCB$M_TIM,UCB$W_STS(R5) ;DISABLE INTERRUPT AND TIMEOUT
983          BISW    #UCB$M_TIMOUT,UCB$W_STS(R5) ;SET DEVICE TIMED OUT
984          SETIPL  UCB$B_DIPL(R5),-                ;Lower IPL to DIPL
985                  ENVIRON=UNIPROCESSOR
986          MOVQ    UCB$L_FR3(R5),R3        ;RETRIEVE SAVED R3 AND R4
987          MOVL    UCB$L_FPC(R5),R2        ;GET SAVED PC
988          CVTWL   -(R2),-(SP)             ;GET OFFSET TO EXCEPTION ROUTINE
989          ADDL    (SP)+,R2                ;CALCULATE ADDRESS OF EXCEPTION ROUTINE
990          JSB     (R2)                    ;CALL EXCEPTION ROUTINE
991 80$:     DEVICEUNLOCK LOCKADDR=UCB$L_DLCK(R5),-
992                      CONDITION=RESTORE  ; Lock interrupt threads/use IPL
993          ASSUME  UCB$B_FIPL  EQ  UCB$B_FLCK
994          ASSUME  SPL$_MIN_INDEX  EQ  32
995          BBC     #5,UCB$B_FLCK(R5),85$   ;Br if not a forklock index
996          FORKUNLOCK LOCK=UCB$B_FLCK(R5)  ;Release FORK threads
997 85$:     SETIPL  #IPL$_TIMER,ENVIRON=UNIPROCESSOR ;Lower IPL to TIMER
998          BRW     40$                     ;
```

```
TIMESCHDL - TIME DEPENDENT SCHEDULING 10-MAY-1989 16:56:06 VAX MACRO V5.0-8 Page 21
X-44U3 SEARCH FOR TIME OUTS 1-NOV-1988 14:42:53 [SYS.SRC]TIMESCHDL.MAR;1 (9)

  999
 1000 ;
 1001 ; TERMINAL READ TIMED IN PROGRESS
 1002 ;
 1003
 1004 ;SMP NOTE READ timeout DUETIME is checked without raising IPL to POWER,
 1005 ;            if the timeout has expired IPL is raised to POWER and
 1006 ;            a readtimeout routine is called
 1007 ;
 1008 90$:     CMPL    UCB$L_TT_RDUE(R5),G^EXE$GL_ABSTIM ;TIME OUT POSSIBLE?
 1009          BGTRU   110$                       ;IF GTRU THEN NO
 1010          SETIPL  #IPL$_POWER,-   ;RAISE TO DEVICE IPL
 1011                  ENVIRON=UNIPROCESSOR
 1012          BBC     #UCB$V_TT_TIMO,UCB$W_DEVSTS(R5),100$ ;IF CLR, TIMEOUT NOT ENABLED
 1013          CMPL    UCB$L_TT_RDUE(R5),G^EXE$GL_ABSTIM ;TIMED OUT?
 1014          BGTRU   100$                       ;IF GTRU THEN NO
 1015          JSB     @UCB$L_TT_RTIMOU(R5)       ;GO OFF TO THE TERMINAL SERVICE
 1016 100$:    SETIPL  #IPL$_TIMER,-              ;LOWER IPL TO THAT OF TIMER
 1017                  ENVIRON=UNIPROCESSOR
 1018 110$:    BRW     50$                        ;
```

```
1020
1021 ;
1022 ; LOG_DISK_QLENS
1023 ;
1024 ; If a MONITOR DISK is active, scan the I/O data base (using all System Blocks),
1025 ; accumulating the current disk queue length value for each mounted disk
1026 ; into UCB$L_QLENACC.
1027 ;
1028 ; Side Effects:
1029 ;
1030 ;       R0 destroyed.
1031 ;
1032
1033 LOG_DISK_QLENS:
1034
1035         BBSC    #0,G^PMS$GL_FLAGS,10$   ; Br if q.l. coll'n active (& clear)
1036                                         ; NOTE -- bit 0 is the active flag
1037         SOBGTR  G^PMS$GL_QLEN_TOCTR,20$ ; Decrement timeout counter ...
1038                                         ; ... and go collect if not timed out
1039         BRB     CHECK_FORK_N_WAIT       ; Collection timed out -- skip past it
1040 10$:
1041         MOVL    G^PMS$GL_QLEN_TOINT, -  ; Re-init the timeout counter
1042                 G^PMS$GL_QLEN_TOCTR
1043         BLEQ    CHECK_FORK_N_WAIT       ; Skip past collection if timeout int 0
1044
1045 ;
1046 ; At this point, we have decided to collect disk queue lengths.
1047 ;
1048
1049 20$:
1050         PUSHR   #^M<R9,R10,R11>         ; Save regs used by this routine
1051         INCL    G^PMS$GL_QLEN_SCANS     ; Count this disk scan
```

```
1053          MOVL    G^SCS$GQ_CONFIG,R11     ; Pick up first system block
1054          BRB     40$                     ; ... and go look for DDBs
1055 30$:
1056          MOVL    SB$L_FLINK(R11),R11     ; Get next system block
1057          CMPL    R11,G^SCS$GQ_CONFIG     ; End of SB chain?
1058          BEQL    100$                    ; Yes, all done
1059 40$:
1060          MOVL    SB$L_DDB(R11),R9        ; Get the first DDB
1061          BEQL    30$                     ; If none, go try next SB
1062          BRB     60$                     ; Got one, go look for UCBs
1063 50$:
1064          MOVL    DDB$L_LINK(R9),R9       ; Get next DDB
1065          BEQL    30$                     ; End of DDB chain, go try next SB
1066 60$:
1067          MOVL    DDB$L_UCB(R9),R10       ; Pick up first UCB
1068          BEQL    50$                     ; If none, go try next DDB
1069
1070 ;
1071 ; At this point, we have a UCB to examine
1072 ;
1073
1074 70$:
1075          CMPB    #DC$_DISK,UCB$B_DEVCLASS(R10) ; Is the unit a disk?
1076          BNEQ    50$                     ; No, skip this entire DDB
1077          BBC     #DEV$V_MNT,UCB$L_DEVCHAR(R10),90$
1078                                          ; Skip UCB if not mounted
1079          BITL    #<DEV$M_CDP+DEV$M_SSM>,UCB$L_DEVCHAR2(R10)
1080                                          ; Class driver path or shad set member?
1081          BNEQ    90$                     ; If so, skip this UCB
1082          BBS     #UCB$V_MNTVERIP,UCB$L_STS(R10),90$
1083                                          ; Skip UCB if mount verify in progress
1084          CVTWL   UCB$W_QLEN(R10),R0      ; Pick up the queue length
1085          BGEQ    80$                     ; Br if pos or zero (as expected)
1086          CLRL    R0                      ; Clear it if negative
1087                                          ; NOTE -- this is a transient condition,
1088                                          ;   which clears itself shortly
1089 80$:
1090          ADDL2   R0,UCB$L_QLENACC(R10)   ; Add q.l. to q.l. accumulator
1091 90$:
1092
1093          MOVL    UCB$L_LINK(R10),R10     ; Get next UCB
1094          BNEQ    70$                     ; ... and go process it
1095          BRB     50$                     ; End of UCB chain, go try next DDB
1096
1097 ;
1098 ; The entire I/O data base has been scanned.
1099 ;
1100
1101 100$:
1102          POPR    #^M<R9,R10,R11>         ; Restore regs used by this routine
1103                                          ; All done, fall thru to next task
```

```
1105
1106 ;
1107 ; PROCESS THE FORK-AND-WAIT WORK QUEUE
1108 ;
1109 ; To avoid an infinite loop of executing new entries placed on the work queue
1110 ; by fork threads resumed from the work queue, the entire work queue is
1111 ; removed from the normal header and hung on a header allocated on the stack.
1112 ; The normal queue header is then initialized thus providing a proper target
1113 ; of new queue insertion operations.
1114 ;
1115         ASSUME  FKB$L_FR4 EQ <FKB$L_FR3 + 4>
1116
1117 CHECK_FORK_N_WAIT:
1118         MOVAL   G^EXE$GL_FKWAITFL, R6   ; Get fork-&-wait queue header address.
1119         ;
1120         ; The following apparent foolish lock of SCS is needed because certain
1121         ; CNXMGR/DSDRIVER threads perform a general REMQUE of blocks without
1122         ; knowing whether the block is on the fork_wait queue or not. Therefore,
1123         ; by synchronizing with these code threads first, we can allow these
1124         ; threads to mis-behave in this manner.
1125         ;
1126         ASSUME  IPL$_TIMER  EQ  IPL$_SCS
1127         LOCK    LOCKNAME=SCS,-          ; Synchronize with SCS threads.
1128                 CONDITION=NOSETIPL,-    ; Don't bother with IPL.
1129                 PRESERVE=NO             ; Don't preserve R0.
1130         LOCK    LOCKNAME=MEGA,-         ; Use high level spinlock.
1131                 PRESERVE=NO             ; Don't preserve R0.
1132         CMPL    (R6), R6                ; Is the queue empty?
1133         BEQL    80$                     ; Branch if queue is empty.
1134         SUBL    #8, SP                  ; Make space on stack for header.
1135         REMQUE  (R6), R5                ; Dequeue the work queue header.
1136         INSQUE  (SP), @4(R5)            ; Queue stack header in its place.
1137         MOVL    R6, (R6)                ; Make real fork=&-wait queue empty.
1138         MOVL    R6, 4(R6)               ;
1139         UNLOCK  LOCKNAME=MEGA,-         ; Unlock high level spinlock.
1140                 NEWIPL=#IPL$_SCS,-      ; Restore our IPL.
1141                 PRESERVE=NO             ; Don't preserve R0.
1142 10$:    REMQUE  @(SP), R5               ; Get an entry from stack work queue.
1143         BVS     90$                     ; Branch if stack work queue empty.
1144         MOVQ    FKB$L_FR3(R5), R3       ; Restore fork context.
1145         MOVZBL  FKB$B_FLCK(R5),R0       ; Get spinlock index.
1146         ASSUME  FKB$B_FIPL  EQ  FKB$B_FLCK
1147         BBC     #5,R0,30$               ; Br if FIPL and not FLCK
1148         PUSHL   R0                      ; Save spinlock index.
1149         JSB     G^SMP$ACQUIRE           ; Acquire the spinlock and raise IPL.
1150         JSB     @FKB$L_FPC(R5)          ; Restart fork thread.
1151         POPL    R0                      ; Restore spinlock index.
1152         JSB     G^SMP$RESTORE           ; Release the spinlock.
1153         SETIPL  #IPL$_SCS,-             ; Restore our IPL.
1154                 ENVIRON=UNIPROCESSOR
1155         BRB     10$                     ; Loop through entire stack work queue.
1156
1157 30$:    ;
1158         ; SMP is not enabled, only use SETIPLs
1159         ;
1160         DSBINT  FKB$B_FIPL(R5),-        ; Establish fork IPL.
1161                 ENVIRON=UNIPROCESSOR
```

```
1162          JSB     @FKB$L_FPC(R5)          ; Restart fork thread.
1163          ENBINT                          ; Restore our IPL.
1164          BRB     10$                     ; Get next packet.
1165
1166 80$:     ;
1167          ; Nothing on FORK_WAIT queue, release lock and continue.
1168          ;
1169          UNLOCK  LOCKNAME=MEGA,-         ; Unlock high level spinlock.
1170                  NEWIPL=#IPL$_SCS,-      ; Restore our IPL.
1171                  PRESERVE=NO             ; Don't preserve R0.
1172          BRB     100$                    ; Continue checking locks.
1173
1174 90$:     ADDL    #8, SP                  ; All done: pop queue header from stack.
1175 100$:    ;
1176          ; Don't bother unlocking SCS! CHECK_LOCKS needs it!
1177          ;
1178 ;;       UNLOCK  LOCKNAME=SCS,-          ; Unlock SCS threads.
1179 ;;               PRESERVE=NO             ; Don't preserve R0.
1180 ;;       JMP     CHECK_LOCKS             ; Then, fall through to checking locks.
1181
1182
1183 ;
1184 ; SCAN FOR WAITING LOCKS THAT MAY HAVE TIMED OUT.  INITIATE A DEADLOCK
1185 ; SEARCH IF ONE IS FOUND.
1186 ;
1187
1188 CHECK_LOCKS:
1189 ;;       LOCK    LOCKNAME=SCS,-          ;LOCK SCS ACCESS
1190 ;;               PRESERVE=NO             ;DON'T PRESERVE R0
1191          MOVAL   G^LCK$GL_TIMOUTQ,R5     ;GET ADDRESS OF LIST HEAD
1192 10$:     MOVL    (R5),R6                 ;GET FIRST ENTRY ON LIST
1193          CMPL    R5,R6                   ;IS LIST EMPTY?
1194          BEQL    20$                     ;YES
1195          CMPL    LKB$L_DUETIME(R6),-     ;NO, HAS THIS ONE TIMED OUT?
1196                  G^EXE$GL_ABSTIM
1197          BGTRU   20$                     ;NO, ALSO NO NEED TO LOOK FURTHER
1198                                          ;AS LIST IS ORDERED
1199          JSB     G^LCK$SEARCHDLCK                ;SEARCH FOR DEADLOCK
1200                                          ;DON'T REPEAT LOOP HERE; IT'S DONE
1201                                          ;IN DEADLOCK
1202 20$:     UNLOCK  LOCKNAME=SCS,-          ;UNLOCK SCS ACCESS
1203                  PRESERVE=NO             ;DON'T PRESERVE R0
1204                                          ;** DON'T CHANGE IPL
1205 ;
1206 ; Invoke the 1 second CPU scheduling processing
1207 ;
1208 SCAN_PROC:
1209          JSB     G^SCH$ONE_SEC
1210
1211 ;
1212 ; DECLARE NON-PAGED AND PAGED DYNAMIC MEMORY AVAILABLE.  THIS IS NECESSARY
1213 ; BECAUSE IN CERTAIN RARE CASES, THE MEMORY ALLOCATION ROUTINES MAY FAIL TO DO
1214 ; THIS AS OFTEN AS NECESSARY.
1215 ; THE MAILBOX RESOURCE IS ALSO DECLARED AVAILABLE FOR SIMILAR REASONS.
1216 ;
1217
1218          MOVZWL  #RSN$_NPDYNMEM,R0       ;NON-PAGED DYNAMIC MEMORY RESOURCE
```

```
1219          JSB     G^SCH$RAVAIL            ;DECLARE RESOURCE AVAILABLE
1220          MOVZWL  #RSN$_PGDYNMEM,R0       ;PAGED DYNAMIC MEMORY RESOURCE
1221          JSB     G^SCH$RAVAIL            ;DECLARE RESOURCE AVAILABLE
1222          MOVZWL  #RSN$_MAILBOX,R0        ;MAILBOX RESOURCE
1223          JSE     G^SCH$RAVAIL            ;DECLARE RESOURCE AVAILABLE
1224          MOVZBL  #RSN$_ASTWAIT,R0        ;ASTWAIT RESOURCE
1225          JSB     G^SCH$RAVAIL            ;DECLARE RESOURCE AVAILABLE
1226
1227 ;
1228 ; ALL DONE - RETURN
1229 ;
1230
1231 TIMEOUT_DONE:
1232          MOVQ    (SP)+,R5                ;RESTORE REGISTERS
1233          RSB                             ;RETURN
1234
1235          .END
```

13 SYSCREPRC.LIS

```
SYSCREPRC CREATE PROCESS SYSTEM SERVICE 10-MAY-1989 16:39:53 VAX MACRO V5.0-8 Page 0
Table of contents

(2)       329        DECLARATIONS
(3)       477        F.XE$CREPRC - CREATE PROCESS SYSTEM SERVICE
(3)       981        ESTABLISH QUOTAS FOR NEW PROCESS
(3)      1235        OVERCHECK - CHECK FOR LEGAL TO EXCEED QUOTA
(3)      1261        PROCESS THE ITEM LIST
(3)      1336        ACTIVATE NEW PROCESS
(3)      1469        ABORT PROCESS CREATION
(3)      1514        MOVSTR - STRING COPY SUBROUTINE
(3)      1574        ALLOCPQB - Allocate PQB from paged pool
```

```
SYSCREPRC CREATE PROCESS SYSTEM SERVICE 10-MAY-1989 16:39:53 VAX MACRO V5.0-8 Page 1
X-32 19-SEP-1988 09:16:28 [SYS.SRC]SYSCREPRC.MAR;1 (2)

     2          .TITLE  SYSCREPRC CREATE PROCESS SYSTEM SERVICE
     3          .IDENT  'X-32'
     4
     5  ;******************************************************************************
     6  ;*                                                                            *
     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984, 1987, 1988 BY                       *
     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
     9  ;*  ALL RIGHTS RESERVED.                                                      *
    10  ;*                                                                            *
    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
    12  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
    15  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
    16  ;*  TRANSFERRED.                                                              *
    17  ;*                                                                            *
    18  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
    19  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
    20  ;*  CORPORATION.                                                              *
    21  ;*                                                                            *
    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
    24  ;*                                                                            *
    25  ;*                                                                            *
    26  ;******************************************************************************
    27
    28  ;++
    29  ; FACILITY: EXECUTIVE, SYSTEM SERVICES
    30  ;
    31  ; ABSTRACT: SYSCREPRC IMPLEMENTS THE CREATE PROCESS SYSTEM SERVICE.
    32  ;
    33  ; ENVIRONMENT: KERNEL MODE
    34  ;
    35  ; AUTHOR: R. HUSTVEDT   , CREATION DATE: 29-NOV-76
    36  ;
    37  ; MODIFIED BY:
    38  ;
    39  ;       X-32    WMC0032         Wayne Cardoza           19-Sep-1988
    40  ;               Add implicit affinity support.
    41  ;
    42  ;       X-31    WMC0031         Wayne Cardoza           22-Aug-1988
    43  ;               New affinity controls.
    44  ;
    45  ;       X-30    WMC0030         Wayne Cardoza           16-Aug-1988
    46  ;               Eliminate requirement for quorum capability.
    47  ;
    48  ;       X-29    JDC0405         Jon Callas              22-JUL-1988
    49  ;               Check duplicate for process names properly.
    50  ;
    51  ;       X-28    WMC0028         Wayne Cardoza           18-May-1988
    52  ;               Eliminate the MOVC3 of the image name just to check its
    53  ;               protection.
    54  ;
    55  ;       X-27    WMC0027         Wayne Cardoza           10-May-1988
    56  ;               Minor performance tweaks.
    57  ;
    58  ;       X-26    CWH5026         CW Hobbs                9-Dec-1987
```

```
;               Avoid reference to other process' PCB after we have
;               rescheduled.
;
;       X-25    JDC0384         Jon Callas              16-NOV-1987
;               Handle failure to allocate a PQB properly.
;
;       X-24    SJF             Stu Farnham             2-Nov-1987
;               Use fat fingers to replace line inadvertently deleted
;               in X-23.
;
;       X-23    SJF             Stu Farnham             26-Oct-1987
;               Use $SETCAP to set QUORUM capability to ensure that
;               proper bookeeping is done.
;
;       X-22    SSA0006         Stan Amway               5-Oct-1987
;               Set QUORUM bit in CREATED process' PCB, not the
;               CREATING process'.
;
;       X-21    JDC0362         Jon Callas              11-SEP-1987
;               Reflect ASTLM back to the PQB.
;               Don't use EXE$ALOPAGWAIT, as it's being removed.
;
;               also includes
;
;               SSA0005         Stan Amway              22-Sep-1987
;               Create all processes requiring QUORUM CPU capability.
;
;       X-20    JWT0298         Jim Teague              27-Aug-1987
;               Parallel processing support.  Add defined bits to
;               $CREPRC STSFLGs so that process can be started up
;               with debugger.
;
;       X-19    JWT0297         Jim Teague              24-Aug-1987
;               Use ADAWI to manipulate PRCCNT; lock down pages and
;               acquire MMG spinlock before deducting shell pages
;               from parent.
;
;       X-18    SSA0004         Stan Amway              10-Aug-1987
;               Swap file allocation changes - don't allocate space here.
;
;               SFxxxxx         Stephen Fiorelli
;               Large working set support.
;
;       X-17    TCM0001         Trudy C. Matthews       09-Jun-1987
;               In the ACTIVATE paragraph, remove the check to see if we
;               are still booting if the attempt to allocte swap
;               space returns a status indicating no page or swap files
;               have been initialized.  Now, page and swap files can be
;               initialized at any point in the life of the system, so checks
;               that assume they are initialized at a particular phase of
;               the bootstrap are in error.  Set the process NOSWAP (the
;               PSWAPM bit in PCB$L_STS) to ensure that it will not be
;               swapped.
;
;       X-16    SSA0003         Stan Amway              13-Apr-1987
;               Call new routine, MMG$DEALLOCSWPAREA, to return swap
;               file space.
```

```
116 ;
117 ;       X-15    JWT0289         Jim Teague              11-Mar-1987
118 ;               JIB lock is now a system spinlock -- remove code
119 ;               to allocate and initialize per-JIB spinlocks.
120 ;
121 ;       X-14    WMC0014         Wayne Cardoza           10-Mar-1987
122 ;               Allow no detached process quota.
123 ;               BYTLM checked agains ORG_BYTLM
124 ;               minor performance tweaks
125 ;               Stay at ASTDEL after process limit chacks.
126 ;
127 ;       X-13    SSA0002         Stan Amway               6-Mar-1987
128 ;               Correctly test return status from MMG$ALLOCSWPAREA.
129 ;
130 ;       X-12    WCT0032         Ward C. Travis           2-Mar-1987
131 ;               Update remaining old lookaside  listhead references
132 ;               to reflect that they are now interlocked queues.
133 ;
134 ;       X-11    ROW0601         Ralph O. Weber          27-FEB-1987 13:16
135 ;               Remove ARB$R_RIGHTSDESC and ARB$R_LOCALRIGHTS hardcoded
136 ;               definitions.  SDL now appears to be generating valid
137 ;               values, and the hardcoded values are no longer correct.
138 ;
139 ;               Also, add several ASSUME statements in the body of code
140 ;               that builds the ARB that is embeded in a PCB.  Several
141 ;               relationships between the PCB and its embeded ARB are
142 ;               assumed but not represented by assume statements.
143 ;
144 ;       X-10    JWT0286         Jim Teague              27-Feb-1987
145 ;               Add code to allocate and initialize JIB spinlock.
146 ;
147 ;       X-9     SF04002         Stephen Fiorelli                05-Feb-1987
148 ;               SCH$GL_NULLPCB becomes SCH$AR_NULLPCB.
149 ;
150 ;       X-8     RNG0008         Rod Gamache             19-Nov-1986
151 ;               Miscellaneous SMP cleanup.
152 ;
153 ;       X-7     RNG0007         Rod Gamache             6-Nov-1986
154 ;               Fix offset from stack for fetching address to store
155 ;               EPID. SMP Poor Man's Lockdown pushes 3 extra longwords
156 ;               onto the stack.
157 ;
158 ;       X-6     SJF             Stu Farnham             5-Nov-1986
159 ;               Initialize PCB$L_AFFINITY.
160 ;
161 ;       X-5     SSA0001         Stan Amway              29-Oct-1986
162 ;               Maintain maximum, concurrent process count.
163 ;
164 ;       X-4     RNG0004         Rod Gamache             21-Oct-1986
165 ;               Use SMP Poor Man's Lockdown macros.
166 ;
167 ;       X-1A3   RNG4003         Rod Gamache             4-Feb-1986
168 ;               Make PQB queue self relative.
169 ;
170 ;       X-1A2   RNG4002         Rod Gamache             30-Jul-1985
171 ;               Add initial SMP support.
172 ;
```

```
;       V04-003 SF04001         Stephen Fiorelli        23-Oct-1985
;               System_service macro used to declare entry point
;               and build a system service descriptor block.
;               Added $SYSVECTORDEF.
;
;       V04-002 WMC0001         Wayne Cardoza           14-Sep-1984
;               Fix LJK0290 to only increment counts for detached processes.
;
;       V04-001 LJK0290         Lawrence J. Kenah       12-Sep-1984
;               Do not count process in SYS$GW_xJOBCNT cell until all error
;               conditions have been tested. This eliminates the need to
;               worry about the xJOBCNT cells along error paths.
;
;       V03-037 ACG0432         Andrew C. Goldstein,    9-Jul-1984  17:27
;               Add PRC$V_NOPASSWORD flag bit, remove DETACH priv
;               requirement from NOUAF, CLISPEC,  and INTER flags.
;               Add and initialize JIB$L_ORG_BYTLM.
;
;       V03-036 PRD0100         Paul R. DeStefano       27-Jun-1984
;               Correct placement of label 10001$ in routine ACTIVATE.
;
;       V03-035 LJK0284         Lawrence J. Kenah       15-May-1984
;               Make SS$_INSSWAPSPACE into a useful error return instead
;               of a system crasher.
;
;       V03-034 MHB0155         Mark Bramhall            1-May-1984
;               Propagate PCB${V|M}_SECAUDIT to new process's PCB.
;
;       V03-033 RAS0297         Ron Schaefer            18-Apr-1984
;               Put back bogus translation of SYS$DISK for compatibility
;               with past history.  Remove KPL0110 and KPL0111.
;
;       V03-032 KPL0111         Peter Lieberwirth       17-Apr-1984
;               The use of IOC$TRNADEVNAM, in V03-029, caused the device
;               string in PQB$T_DISK not to have a trailing colon.  This
;               caused PROCSTRT to create an invalid translation for
;               SYS$DISK.  Fix here by re-appending the colon.
;
;       V03-031 MHB0136         Mark Bramhall           12-Apr-1984
;               Add support for PRC$V_CLISPEC.
;
;       V03-030 MHB0134         Mark Bramhall           10-Apr-1984
;               Creators w/o JIBs => Username = SYSTEM, Account = binary nulls.
;               Account names with a leading binary null byte are special:
;                       <0><0>... => <0><0>...  All binary nulls stays as is.
;                       <0><x>... => <x>...< >  Others are shifted left one.
;               Move new spawn CLI information to PQB from P1 space.
;
;       V03-029 KPL0110         Peter Lieberwirth       31-Mar-1984
;               Translate SYS$DISK by calling G^IOC$TRANDEVNAM, which
;               uses $TRNLNM.  This call replaces obsolete $TRNLOG use.
;
;       V03-028 SRB0119         Steve Beckhardt         26-Mar-1984
;               Another round in the broken branch grame.
;
;       V03-027 LJK0263         Lawrence J. Kenah       29-Feb-1984
;               We're playing the broken branch game again. The various calls
```

```
230 ;               to G^EXE$DEANONPAGED need more than word displacement.
231 ;
232 ;       V03-026 MH0002          Hai Huang               1-Feb-1984
233 ;               Add job-wide mount support, i.e. initialize mount listhead
234 ;               in JIB.
235 ;
236 ;       V03-025 LJK0259         Lawrence J. Kenah       23-Jan-1984
237 ;               Fix incorrect register usage bug in PQB deallocation.
238 ;
239 ;       V03-024 LJK0258         Lawrence J. Kenah       18-Jan-1984
240 ;               Fix bug introduced by LJK0257. Make JIB creation code handle
241 ;               the case of the swapper, a process that does not own a JIB.
242 ;
243 ;       V03-023 ACG0385         Andrew C. Goldstein,    11-Jan-1984  18:39
244 ;               Make MAXDETACH and MAXJOBS JIB fields words
245 ;
246 ;       V03-022 LJK0257         Lawrence J. Kenah       21-Dec-1983
247 ;               Make changes to support larger PQB. Remove support for ACCOUNT
248 ;               and USERNAME fields in P1 space. Use $TRNLNM to pick up
249 ;               translation of SYS$DISK. Perform general cleanup.
250 ;
251 ;       V03-021 TMK0001         Todd M. Katz            12-Oct-1983
252 ;               Add JTQUOTA (job-wide logical name table creation quota)
253 ;               to the process quota block and as a quota in the quota list
254 ;               for G^SYS$CREPRC. No special processing is required for this
255 ;               new quota item.
256 ;
257 ;       V03-020 JWT0138         Jim Teague              11-Oct-1983
258 ;               Fix broken branch to G^SCH$CHSE.
259 ;
260 ;       V03-019 RAS0181         Ron Schaefer            05-Sep-1983
261 ;               Convert creation of SYS$INPUT, SYS$OUTPUT, SYS$ERROR,
262 ;               and SYS$DISK logical names to use $CRELNM.
263 ;
264 ;       V03-018 WMC0008         Wayne Cardoza           01-Aug-1983
265 ;               New item list codes for logical name attributes.
266 ;
267 ;       V03-017 ACG0347         Andrew C. Goldstein,    1-Aug-1983  13:21
268 ;               Fix register use bug in ACG0335
269 ;
270 ;       V03-016 WMC0007         Wayne Cardoza           28-Jul-1983
271 ;               Move bumping of interactive and batch counts here.
272 ;
273 ;       V03-015 WMC0006         Wayne Cardoza           05-JUL-1983
274 ;               No privilege needed for interactive subprocess.
275 ;
276 ;       V03-014 CWH1010         CW Hobbs                20-Jun-1983
277 ;               Add a comment which calls attention to an order dependency
278 ;               in the call to G^EXE$IPID_TO_EPID.
279 ;
280 ;       V03-013 ACG0335         Andrew C. Goldstein,    9-May-1983  16:26
281 ;               Propagate default file protection in PCB; copy extended
282 ;               rights list to created process.
283 ;
284 ;       V03-012 WMC0005         Wayne Cardoza           27-Apr-1983
285 ;               Change PSECT of detach code.
286 ;               SS$_INSSWAPSPACE for no swap space.
```

```
;
;       V03-011 WMC0004         Wayne Cardoza           14-Apr-1983
;               New STSFLG flags.
;
;       V03-010 WMC0003         Wayne Cardoza           31-Mar-1983
;               Second half of the detach changes.
;
;       V03-009 WMC0002         Wayne Cardoza     -     10-Mar-1983
;               Liberalized rules on creation of detached processes.
;
;       V03-008 ACG0318         Andrew C. Goldstein,    8-Mar-1983  20:27
;               Initialize new ARB fields in created process
;
;       V03-007 MTR0001         Michael T. Rhodes       28-Feb-1983
;               Change the privilege requirements for setting the initial
;               process state flags to require only the creator to have
;               privilege.
;
;       V03-006 CWH1002         CW Hobbs                24-Feb-1983
;               Create new extended process ident PCB$L_EPID.  Return the
;               extended pid to the pidadr argument if specifed.  The
;               extended pid of a subprocess owner propagates to the
;               new PCB$L_EOWNER field.
;
;       V03-005 CWH1001         CW Hobbs                15-Feb-1983
;               Change from sequential to round-robin PIX allocation.
;
;       V03-004 WMC0001         Wayne Cardoza           18-Oct-1982
;               Add support for item list argument for page file control.
;
;       V03-003 LJK48272        Lawrence J. Kenah       10-Aug-1982
;               Insure that PCB$L_JIB is clear before allocating PQB
;               in case PQB allocation fails. Error code assumes that
;               contents of JIB field are always valid if nonzero.
;               Remove $PRDEF and $SSDEF calls.
;
;       V03-002 LJK0169         Lawrence J. Kenah       2-Jun-1982
;               Insure that revised CPU limit is stored in PQB along error
;               paths. Use ROTL to perform unsigned divide by two.
;
;--
```

```
329          .SBTTL  DECLARATIONS
330 ;
331 ; INCLUDE FILES:
332 ;
333          $ACLDEF                         ; DEFINE ACL BLOCK
334          $ARBDEF                         ; DEFINE ACCESS RIGHTS BLOCK
335          $CPBDEF                         ; CAPABILITY DEFINITIONS
336          $DYNDEF                         ; DATA STRUCTURE IDENTIFIERS
337          $IPLDEF                         ; DEFINE INTERRUPT PRIORITY LEVELS
338          $JIBDEF                         ; DEFINE JOB INFORMATION BLOCK
339          $LNMDEF                         ; DEFINE LNM OFFSETS
340          $LNMSTRDEF                      ; DEFINE LNM BLOCK OFFSETS
341          $PCBDEF                         ; DEFINE PCB OFFSETS
342          $PFNDEF                         ; DEFINE PFN CONSTANTS
343          $PHDDEF                         ; DEFINE PHD OFFESTS
344          $PQBDEF                         ; DEFINE PROCESS QUOTA BLOCK
345          $PQLDEF                         ; DEFINE PROCESS QUOTA LIST CODES
346          $PRCDEF                         ; DEFINE $CREPRC STATUS FLAGS
347          $PRIDEF                         ; DEFINE PRIORITY INCREMENT CLASSES
348          $PRVDEF                         ; DEFINE PRIVILEGE BITS
349          $SYSVECTORDEF                   ; DEFINE SYSTEM SERVICE VECTOR OFFSETS
350
351 ;
352 ; MACROS:
353 ;
354
355 ;
356 ; MACRO TO CREATE STSFLG MAPPING AND PRIVILEGE CHECK TABLES:
357 ;       STSNAM          OPTIONAL STATUS BIT NAME TO SET IN PCB
358 ;       PRVNAM          OPTIONAL REQUIRED PRIVILEGE BIT NAME
359 ;       NOSUBPRV        OPTIONAL NO PRIVILEGE REQUIRED IF SUBPROCESS FLAG
360 ;
361          .MACRO  STSFLAG STSNAM,PRVNAM,NOSUBPRV
362          .IF     B,PRVNAM
363          .BYTE   -1
364          .IFF
365          .IF     B,NOSUBPRV
366          .BYTE   PRV$V_'PRVNAM
367          .IFF
368          .BYTE   PRV$V_'PRVNAM ! ^X80
369          .ENDC
370          .ENDC
371          .IF     B,STSNAM
372          .BYTE   -1
373          .IFF
374          .BYTE   PCB$V_'STSNAM
375          .ENDC
376          .ENDM   STSFLAG
377
378 ;
379 ; MACRO TO CALL STRING MOVING AND VERIFICATION ROUTINE
380 ;
381          .MACRO  MOVSTRING LIM=15,SRC,DST
382          BSBW    MOVSTR                  ; CALL MOVE SUBROUTINE
383          .BYTE   LIM                     ; COUNT LIMIT
384          .BYTE   SRC@-2                  ; SOURCE OFFSET FROM AP
385          .IF IDN  <DST> <#0>
```

```
386          .WORD 0
387          .IFF
388          .WORD   PQB$T_'DST              ; DESTINATION OFFSET IN PQB
389          .ENDC
390          .ENDM   MOVSTRING
391
392 ;
393 ; EQUATED SYMBOLS:
394 ;
395
396 PIDADR=4                                 ; PID ADDRESS
397 IMAGE=8                                  ; IMAGE NAME
398 INPUT=12                                 ; INPUT LOGICAL NAME DESCRIPTOR
399 OUTPUT=16                                ; OUTPUT LOGICAL NAME DESCRIPTOR
400 ERROR=20                                 ; ERROR LOGICAL NAME DESCRIPTOR
401 PRVADR=24                                ; PRIVILEGE MASK ADDRESS
402 QUOTA=28                                 ; QUOTA BUFFER POINTER
403 PRCNAM=32                                ; PROCESS NAME DESCRIPTOR
404 BASPRI=36                                ; BASE PRIORITY
405 UIC=40                                   ; UIC
406 MBXUNT=44                                ; MAILBOX UNIT NUMBER
407 STSFLG=48                                ; STATUS FLAG MASK
408 ITMLST=52                                ; ITEM LIST
409
410 PQL_V_DEDUCT=0                           ; DEDUCTIBLE QUOTA FLAG
411 ITMLST_ARG=13                            ; ARGUMENT NUMBER FOR THE ITEM LIST
412
413 CURPCB = -4                              ; OFFSET FROM FP TO SAVED R4
414
415 ;
416 ; OWN STORAGE:
417 ;
418
419          DECLARE_PSECT   EXEC$PAGED_DATA ; PAGEABLE PSECT
420
421          ASSUME  PRC$V_SSRWAIT  EQ  0
422          ASSUME  PRC$V_SSFEXCU  EQ  1
423          ASSUME  PRC$V_PSWAPM   EQ  2
424          ASSUME  PRC$V_NOACNT   EQ  3
425          ASSUME  PRC$V_BATCH    EQ  4
426          ASSUME  PRC$V_HIBER    EQ  5
427          ASSUME  PRC$V_NOUAF    EQ  6
428          ASSUME  PRC$V_NETWRK   EQ  7
429          ASSUME  PRC$V_DISAWS   EQ  8
430          ASSUME  PRC$V_DETACH   EQ  9
431          ASSUME  PRC$V_INTER    EQ  10
432          ASSUME  PRC$V_IMGDMP   EQ  11
433          ASSUME  PRC$V_CLISPEC  EQ  12
434          ASSUME  PRC$V_NOPASSWORD EQ  13
435          ASSUME  PRC$V_DEBUG    EQ  14
436          ASSUME  PRC$V_DBGTRU   EQ  15
437
438 STSFLGTBL:                              ; TRANSLATION TABLE FOR STATUS FLAG BITS
439          STSFLAG SSRWAIT                ; BIT 0 => RESOURCE WAIT
440          STSFLAG SSFEXCU                ; BIT 1 => SYSTEM SERVICE FAIL EXCEPTION
441                                         ;          FOR USER MODE
442          STSFLAG PSWAPM,PSWAPM          ; BIT 2 => PROCESS SWAP MODE
```

```
443          STSFLAG NOACNT,NOACNT           ; BIT 3 => NO ACCOUNTING MESSAGE
444          STSFLAG BATCH,DETACH            ; BIT 4 => BATCH
445          STSFLAG HIBER                   ; BIT 5 => HIBERNATE BEFORE CALLING
446                                          ;          INITIAL IMAGE IN PROCSTRT
447          STSFLAG LOGIN                   ; BIT 6 => LOGIN WITHOUT READING AUTH FILE
448          STSFLAG NETWRK,DETACH           ; BIT 7 => NETWORK
449          STSFLAG DISAWS                  ; BIT 8 => DISABLE WORKING SET ADJUST
450          STSFLAG                         ; BIT 9 => DETACH
451          STSFLAG INTER                   ; BIT 10 => INTERACTIVE
452          STSFLAG                         ; BIT 11 => IMAGE DUMP
453          STSFLAG                         ; BIT 12 => PASS ON CLI SPECIFICATIONS
454          STSFLAG                         ; BIT 13 => NO USERNAME DIALOGUE
455          STSFLAG                         ; BIT 14 => CLI DEBUG flag
456          STSFLAG                         ; BIT 15 => CLI DBGTRU flag
457 STSFLGCNT=<.-STSFLGTBL>@-1                ; NUMBER OF STATUS FLAGS
458
459 ; THE FOLLOWING TEXT FIELDS ARE USED WHEN THE CREATING PROCESS (SUCH AS THE
460 ; SWAPPER) DOES NOT HAVE A JIB.
461
462 DEFAULT_NAMES:
463          .ASCII  'SYSTEM'                ; Username is SYSTEM, blank padded
464          .BYTE   ^A' ' [JIB$S_USERNAME - <. - DEFAULT_NAMES>]
465          .BYTE   0 [JIB$S_ACCOUNT]       ; Account name is binary nulls
466          ASSUME  <. - DEFAULT_NAMES> EQ <JIB$S_USERNAME + JIB$S_ACCOUNT>
467
468 ; LOGICAL NAME DATA FOR USE IN TRANSLATING SYS$DISK
469
470 LNM_TBL:.ASCID  \LNM$FILE_DEV\
471 LNM_ATTR = ^X0103
472
473
```

```
475          DECLARE_PSECT   EXEC$PAGED_CODE
476
477          .SBTTL  EXE$CREPRC - CREATE PROCESS SYSTEM SERVICE
478 ;++
479 ; FUNCTIONAL DESCRIPTION:
480 ;       EXE$CREPRC CREATES A NEW PROCESS ACCORDING TO THE
481 ;       SUPPLIED PARAMETERS.  THE NEW PROCESS MAY BE EITHER A SUB-PROCESS
482 ;       OR AN INDEPENDENT, DETACHED PROCESS.
483 ;
484 ; CALLING SEQUENCE:
485 ;       CALLG   ARGLIST,G^EXE$CREPRC
486 ;
487 ; INPUT PARAMETERS:
488 ;       PIDADR(AP) - ADDRESS AT WHICH TO RETURN PID OF CREATED PROCESS
489 ;       IMAGE(AP)  - ADDRESS OF IMAGE NAME STRING DESCRIPTOR
490 ;       INPUT(AP)  - ADDRESS OF INPUT NAME STRING DESCRIPTOR
491 ;       OUTPUT(AP) - ADDRESS OF OUTPUT NAME STRING DESCRIPTOR
492 ;       ERROR(AP)  - ADDRESS OF ERROR LOGICAL NAME STRING DESCRIPTOR
493 ;       PRVADR(AP) - ADDRESS OF PRIVILEGE MASK FOR CREATED PROCESS
494 ;       QUOTA(AP)  - POINTER TO QUOTA BUFFER
495 ;       PRCNAM(AP) - ADDRESS OF PROCESS NAME STRING DESCRIPTOR
496 ;       BASPRI(AP) - BASE PRIORITY FOR CREATED PROCESS
497 ;       UIC(AP)    - UIC FOR CREATED PROCESS(0 => SUB-PROCESS)
498 ;       MBXUNT(AP) - MAILBOX UNIT NUMBER FOR TERMINATION MESSAGES
499 ;       STSFLG(AP) - STATUS FLAG SETTINGS FOR CREATED PROCESS
500 ;       ITMLST(AP) - ITEM LIST
501 ;       R4 - ADDRESS OF CURRENT PROCESS CONTROL BLOCK
502 ;
503 ;                          BIT     MEANING
504 ;                          ---     -------
505 ;                           0      RESOURCE WAIT DISABLE
506 ;                           1      SYSTEM SERVICE FAIL EXCEPTION ENABLE
507 ;                           2      PROCESS SWAP MODE
508 ;                           3      ACCOUNTING MESSAGE DISABLE
509 ;                           4      BATCH INDICATOR
510 ;                           5      HIBERNATE BEFORE CALLING INITIAL IMAGE
511 ;                           6      BYPASS LOGIN VERIFICATION FOR DETACHED
512 ;                                  PROCESS.
513 ;                           7      NETWORK INDICATOR
514 ;                           8      DISABLE WORKING SET ADJUSTMENT
515 ;                           9      DETACHED PROCESS
516 ;                          10      INTERACTIVE INDICATOR
517 ;                          11      IMAGE DUMP ON FATAL ABORT
518 ;                          12      PASS ON CLI SPECIFICATIONS
519 ;
520 ;
521 ; OUTPUT PARAMETERS:
522 ;       R0  -  COMPLETION STATUS CODE
523 ;       @PIDADR(AP) - PROCESS ID (PID) OF CREATED PROCESS
524 ;
525 ; COMPLETION CODES:
526 ;       SS$_NORMAL    -   SUCCESSFUL COMPLETION
527 ;       SS$_ACCVIO    -   ACCESS VIOLATION
528 ;       SS$_DUPLNAM   -   DUPLICATE PROCESS NAME
529 ;       SS$_EXQUOTA   -   EXCEEDED QUOTA
530 ;       SS$_INSFMEM   -   INSUFFICIENT MEMORY AVAILABLE
531 ;       SS$_IVLOGNAM  -   INVALID LOGICAL NAME
```

```
;       SS$_IVQUOTAL -   INVALID QUOTA LIST
;       SS$_IVSTSFLG -   INVALID STATUS FLAG ARGUMENT
;       SS$_NOPRIV   -   NO PRIVILEGE FOR SPECIFIED OPERATION
;
; SIDE EFFECTS:
;       IF NO ERRORS ARE DETECTED, A NEW PROCESS WILL HAVE BEEN ACTIVATED
;       AND MARKED NON-RESIDENT.  THE INITIAL INSWAP FOR THIS PROCESS
;       WILL BE FROM THE SHELL PROCESS.  EXECUTION FOR THIS PROCESS
;       BEGINS IN THE ROUTINE G^EXE$PROCSTRT WHICH WILL MOVE THE INFORMATION
;       FROM THE PROCESS QUOTA BLOCK TO THE APPROPRIATE LOCATIONS
;       IN THE PROCESS CONTEXT.  CONTROL WILL THEN BE GIVEN TO THE
;       SPECIFIED IMAGE.
;
;--

        .ENABL  LSB
        SYSTEM_SERVICE  CREPRC,-
                        <R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>,-
                        MODE=KERNEL,-
                        NARG=12

        PUSHL   R4                      ; SAVE CREATOR'S PCB ADDRESS
        PUSHL   PIDADR(AP)              ; SAVE PID RETURN ADDRESS
        BEQL    10$                     ; NONE, NO PROBE
        IFNOWRT #4,@(SP),20$            ; CHECK FOR WRITABLE
10$:    MOVL    UIC(AP),R11             ; FETCH UIC FOR CREATED PROCESS
        BEQL    29$                     ; NOT SPECIFIED
        CMPL    R11,PCB$L_UIC(R4)       ; IS IT SAME UIC AS CREATOR
        BEQL    30$                     ; NO PRIVILEGE NEEDED
        IFPRIV  DETACH,30$              ; FULL DETACH PRIVILEGE
        IFPRIV  CMKRNL,30$              ; OR CHANGE MODE TO KERNEL
        MOVZWL  #SS$_NOPRIV,R0          ; NO, SET ERROR CODE
15$:    RET                             ; RETURN

20$:    MOVZWL  #SS$_ACCVIO,R0          ; SET ERROR CODE FOR ACCESS VIOLATION
        RET                             ; RETURN

; FAST ALLOCATION OF PQB FAILED. DO IT THE HARD WAY

26$:    BSBW    ALLOCPQB                ; ALLOCATE ANOTHER PQB
        MOVL    R2,R9                   ; It always succeeds
        BRB     50$                     ;

510$:   BRW     51$                     ; Branch helper

29$:    BBC     #PRC$V_DETACH,STSFLG(AP),30$
        MOVL    PCB$L_UIC(R4),R11       ; DETACHED PROCESS (SAME UIC) REQUESTED
30$:    JSB     G^EXE$ALLOCPCB          ; ALLOCATE PCB, WAIT IF NECESSARY
        BLBC    R0,15$                  ; RETURN ERROR IF FAILURE
        MOVL    R2,R10                  ; SAVE ADDRESS OF NEW PCB
        MOVC5   #0,(SP),#0,#<PCB$C_LENGTH-12>,12(R10)    ; CLEAR PCB
        MOVL    CURPCB(FP),R4           ; RESTORE CREATOR PCB ADDRESS
        BICL3   #^C<PCB$M_SECAUDIT>,-   ; PROPAGATE (ONLY) MANDATORY AUDITING
                PCB$L_STS(R4),PCB$L_STS(R10) ;  FROM PARENT TO NEW PCB
        $REMQHI G^EXE$GQ_PQBIQ,R9       ; ATTEMPT FAST PQB ALLOCATION
                                        ; NOTE: R0 is scratch in $REMQHI macro
        BVS     26$                     ; OUT OF LINE IF FAILURE
```

```
589 ;
590 ;       INITIALIZE NEW PCB
591 ;
592 50$:    TSTL    R11                     ; IS THIS A DETACHED CREATE?
593         BNEQ    510$                    ; YES, GO ALLOCATE A NEW JIB
594         MOVL    PCB$L_JIB(R4),R2        ; GET JIB ADDRESS OF PARENT
595         MOVL    R2,PCB$L_JIB(R10)       ; SAVE POINTER TO JIB IN OFFSPRING
596         ADAWI   #1,JIB$W_PRCCNT(R2)     ; ADD ANOTHER SUBPROCESS
597         PMLREQ  END=2300$               ; Lock pages

    ;++
    ; NB: Co-routine address + 2 LWs have been placed on top of stack
    ;--
598         LOCK    LOCKNAME=MMG,-          ; Grab MMG spinlock
599                 SAVIPL=-(SP)
600         SUBL    #SWP$C_SHELLPFIL,JIB$L_PGFLCNT(R2); CHARGE FOR SHELL PAGES
601         BLSS    23$                     ; BR IF OUT OF PAGE FILE QUOTA
602         UNLOCK  LOCKNAME=MMG,-
603                 NEWIPL=(SP)+,-
604                 PRESERVE=NO
605         PMLEND                          ; Through with locked pages

    ;++
    ; NB: Co-routine address + 2 LWs have been removed from top of stack
    ;--
606         CMPW    JIB$W_PRCCNT(R2),JIB$W_PRCLIM(R2) ; OVER LIMIT?
607         BLEQU   56$                     ; IF LEQU NO, CONTINUE
608         BRB     2300$                   ; Skip unlock (already done)
609
610 23$:    UNLOCK  LOCKNAME=MMG,-          ; Release MMG
611                 NEWIPL=(SP)+,-
612                 PRESERVE=NO
613         PMLEND

    ;++
    ; NB: Co-routine address + 2 LWs have been removed from top of stack
    ;--
614 2300$:  MOVZWL  #SS$_EXQUOTA,R0         ; SET ERROR STATUS
615 24$:    BRW     ABORT                   ; YES ABORT
616
617 51$:    JSB     G^EXE$ALLOCJIB          ; ALLOCATE JIB FOR MASTER PROCESS
618         BLBC    R0,24$                  ; BR IF NO SPACE AVAILABLE
619         MOVAL   JIB$L_MTLFL(R2),JIB$L_MTLFL(R2) ; INITIALIZE MOUNT LISTHEAD
620         MOVAL   JIB$L_MTLFL(R2),JIB$L_MTLBL(R2) ;
621         MOVL    R2,PCB$L_JIB(R10)       ; SAVE POINTER TO JIB
622
623 ; THE USERNAME AND ACCOUNT FIELDS OF THE CREATOR'S JIB ARE MOVED INTO THE
624 ; NEW JIB. THE REST OF THE JIB IS CLEARED. THE TWO ASSUME STATEMENTS INSURE
625 ; THAT THE JIB LAYOUT ALLOWS ALL OF THESE SHENANIGANS TO BE ACCOMPLISHED
626 ; WITH A SINGLE MOVC5 INSTRUCTION.
627
628         ASSUME JIB$T_USERNAME EQ 12
629         ASSUME JIB$T_ACCOUNT EQ <12 + JIB$S_USERNAME>
630
631         MOVL    PCB$L_JIB(R4),R3        ; GET JIB ADDRESS OF CREATOR
632         BNEQ    53$                     ; JIB EXISTS, GO USE IT
633
```

```
634 ; THE SWAPPER PROCESS DOES NOT HAVE A JIB. LOAD R3 WITH THE ADDRESS OF SOME
635 ; TEXT FIELDS THAT LOAD THE NEW JIB WITH A DEFAULT USERNAME AND ACCOUNT.
636
637         MOVAB   DEFAULT_NAMES-JIB$T_USERNAME,R3 ; GET STRING ADDRESS
638 53$:    MOVC5   #<JIB$S_USERNAME + JIB$S_ACCOUNT>,-
639                 JIB$T_USERNAME(R3),-    ; CREATOR'S JIB IS SOURCE
640                 #0,-                    ; FILL THE REST WITH ZEROS
641                 #<JIB$C_LENGTH-12>,-
642                 JIB$T_USERNAME(R2)      ; NEW JIB IS DESTINATION
643
644 ; Check the account name for a leading binary null.  Account names with
645 ; a leading binary null are special, reserved to DIGITAL, account names.
646 ; The following account name combinations are possible:
647 ;       <x>...          Normal account name; left as is.
648 ;       <0><0>...       Initial startup account name; left as is.
649 ;       <0><x>...       Special account name; shifted left one place.
650
651         MOVL    PCB$L_JIB(R10),R2       ; JIB in R2 again
652         TSTB    JIB$T_ACCOUNT(R2)       ; A leading binary null byte?
653         BNEQ    56$                     ; Nope, leave account name as is
654         TSTB    JIB$T_ACCOUNT+1(R2)     ; Anything after the null to shift?
655         BEQL    56$                     ; Nope, leave account name alone
656         MOVC5   #JIB$S_ACCOUNT-1,-      ; Move all but the first byte
657                 JIB$T_ACCOUNT+1(R2),-   ;  of the account name,
658                 #^A' ',-                ;   padding with a blank,
659                 #JIB$S_ACCOUNT,-        ;    into the full
660                 JIB$T_ACCOUNT(R2)       ;     account name
661
662 56$:    MOVAL   PCB$L_ASTQFL(R10),PCB$L_ASTQFL(R10)     ; SET UP AST Q HEADER
663         MOVAL   PCB$L_ASTQFL(R10),PCB$L_ASTQBL(R10)     ;
664         MOVB    #^X0F,PCB$B_ASTEN(R10)                  ; SET ALL AST ENABLES
665         MOVAL   PCB$L_LOCKQFL(R10),PCB$L_LOCKQFL(R10)   ; SET UP LOCK Q HEADER
666         MOVAL   PCB$L_LOCKQFL(R10),PCB$L_LOCKQBL(R10)
667 ;
668 ;       BUILD THE ARB THAT IS EMBEDED IN THE PCB
669 ;
670 ;       The following code makes several assumptions regarding the ARB
671 ;       that is embeded in the PCB.  Most important amoung these being
672 ;       the assumption that PCB$L_UIC falls in the correct ARB location.
673 ;       The following assume statement attempt to verify these assumptions
674 ;       in the code.
675 ;
676 $PCBARB = PCB$R_PCBARB
677         ASSUME  <PCB$Q_PRIV - $PCBARB> EQ ARB$Q_PRIV    ; PRIVS IN SAME PLACE?
678         ASSUME  PCB$S_PCBARB GE ARB$C_LENGTH            ; PCBARB MUST BE BIG ENOUGH
679         ASSUME  <PCB$L_UIC - $PCBARB> EQ ARB$L_UIC      ; UIC MUST BE IN THE
680         ASSUME  ARB$L_UIC EQ ARB$R_LOCALRIGHTS          ; RIGHT PLACE
681
682         MOVL    CURPCB(FP),R4                           ; RESTORE CREATOR PCB ADDRES
683         MOVL    $PCBARB+ARB$L_RIGHTSLIST+8(R4),R6       ; GET EXTENDED RIGHTS LIST,
684         MOVL    PCB$L_DEFPROT(R4),PCB$L_DEFPROT(R10)    ; COPY DEFAULT FILE PROTECTI
685         MOVL    G^SCH$GL_DEFAULT_PROCESS_CAP,R0
686         MOVL    R0,PCB$L_PERMANENT_CAPABILITY(R10)      ; INITIALIZE CAPABILITY REQU
687         MOVL    R0,PCB$L_CAPABILITY(R10)
688         MOVB    G^SCH$GL_AFFINITY_SKIP,-                ; IMPLICIT AFFINITY SKIP
689                 PCB$B_AFFINITY_SKIP(R10)
690         BBC     #CPB$V_IMPLICIT_AFFINITY,R0,565$        ; IMPLICIT AFFINITY?
```

```
691          MOVL    PCB$L_AFFINITY(R4),PCB$L_AFFINITY(R10)  ; PROPAGATE FROM PARENT
692 565$:    BBC     #31,SP,57$                              ; BRANCH IF NOT INITIAL CREA
693          MOVZWL  G^SYS$GW_FILEPROT,PCB$L_DEFPROT(R10)    ; SET SYSTEM FILE PROTECTION
694 57$:     MOVC3   #ARB$C_LENGTH,$PCBARB(R4),$PCBARB(R10)  ; COPY ENTIRE ARB
695          MOVAB   $PCBARB(R10),PCB$L_ARB(R10)             ; SET ADDRESS OF ARB
696          MOVAB   $PCBARB+ARB$R_RIGHTSDESC(R10),-
697                  $PCBARB+ARB$L_RIGHTSLIST(R10)           ; ADDR OF LOCAL RIGHTS DESC
698          MOVAB   $PCBARB+ARB$R_LOCALRIGHTS(R10),-
699                  $PCBARB+ARB$R_RIGHTSDESC+4(R10)         ; ADDR OF LOCAL RIGHTS LIST
700          TSTL    R6                                      ; SEE IF EXTENDED RIGHTS LIS
701          BEQL    59$                                     ; BRANCH IF NONE
702          CLRL    $PCBARB+ARB$L_RIGHTSLIST+8(R10) ; MARK NONE ALLOCATED YET
703          MOVZWL  ACL$W_SIZE(R6),R1                       ; GET SIZE
704          JSB     G^EXE$ALLOCBUF                          ; AND ALLOCATE A NEW ONE
705          BLBS    R0,58$                                  ; KEEP GOING IF SUCCESS
706          BRW     ABORT                                   ; ABORT CREATION
707
708 575$:    BRW     ACCVIO                  ; GIVE ACCESS VIOLATION AND ABORT
709
710 58$:     MOVL    R2,R7                                   ; SAVE ADDRESS
711          MOVL    R2,$PCBARB+ARB$L_RIGHTSLIST+8(R10)      ; STORE IN PCB
712          MOVC3   R1,(R6),(R7)                            ; COPY RIGHTS LIST CONTENTS
713          MOVAB   12(R7),4(R7)                            ; SET DESCRIPTOR POINTER
714 59$:     MOVW    MBXUNT(AP),PCB$W_TMBU(R10)              ; TERMINATION MAILBOX UNIT
715          MOVL    G^SWP$GL_SHELLSIZ,PCB$L_PPGCNT(R10)     ; AND PROCESS PAGE COUNT
716 ;
717 ;        INITIALIZE QUOTA BUFFER
718 ;
719          MOVC5   #0,(SP),#0,#<PQB$C_LENGTH-12>,12(R9)    ; CLEAR QUOTA BUFFER
720 ;
721          MOVSTRING       LIM=15,-        ; VALIDATE PROCESS NAME
722                  SRC=PRCNAM,-            ; FROM PROCESS NAME DESCRIPTOR
723                  DST=#0                  ; WITHOUT MOVING IT
724          TSTL    R2                      ; WAS A PROCESS NAME SPECIFIED
725          BEQL    60$                     ; NO PROCESS NAME, NULL STRING
726          MOVZWL  (R2),R6                 ; GET LENGTH
727          BEQL    60$                     ; NO PROCESS NAME, NULL STRING
728          MOVB    R6,PCB$T_LNAME(R10)     ; LENGTH
729          MOVC3   R6,@4(R2),PCB$T_LNAME+1(R10)    ; SET INTO NEW PCB
730 60$:     MOVL    CURPCB(FP),R4           ; RESTORE PCB ADDRESS
731          MOVQ    @PCB$L_PHD(R4),R2       ; GET PRIVILEGE MASK
732          MOVL    PRVADR(AP),R7           ; FETCH ADDRESS OF PRIVILEGE MASK
733          BEQL    70$                     ; NONE SPECIFIED
734          MOVQ    R2,R0                   ; SAME PRIVILEGES
735          IFNORD  #8,(R7),575$            ; ERROR IF NOT READABLE
736          MOVQ    (R7),R2                 ; FETCH NEW PRIVILEGE MASK
737          IFPRIV  SETPRV,70$              ; SKIP MINIMIZATION IF SETPRV
738          MCOML   R0,R0                   ; INVERT FOR BIT CLR
739          MCOML   R1,R1                   ;
740          BICL    R0,R2                   ; MINIMIZE PRIVILEGES
741          BICL    R1,R3                   ; IF CALLER IS NOT PRIVILEGED ENOUGH
742 70$:     MOVQ    R2,PQB$Q_PRVMSK(R9)     ; SET PRIVILEGES IN BUFFER
743          BICL3   #^C31,BASPRI(AP),R0     ; FETCH DESIRED BASE PRIORITY
744          SUBL3   R0,#31,R0               ; CONVERT TO INTERNAL PRIORITY
745          IFPRIV  SETPRI,100$             ; SKIP MINIMIZATION IF PRIVILEGE
746          CMPB    R0,PCB$B_PRIB(R4)       ; OTHERWISE MINIMIZE WITH CALLERS BASE
747          BGEQU   100$                    ; GOOD VALUE, USE IT
```

```
748          MOVB    PCB$B_PRIB(R4),R0       ; NO, USE CALLERS BASE PRIORITY
749 100$:    MOVB    R0,PCB$B_PRIB(R10)      ; SET IN NEW PCB
750          MOVB    R0,PCB$B_PRI(R10)       ; AS BOTH CURRENT AND BASE PRIORITY
751          MOVL    R11,PCB$L_UIC(R10)      ; STORE UIC FOR NEW PROCESS
752          BNEQ    110$                    ; BR IF IT WAS SPECIFIED
753          MOVL    PCB$L_UIC(R4),PCB$L_UIC(R10)    ; OTHERWISE USE UIC OF CREATOR
754          MOVL    PCB$L_PID(R4),PCB$L_OWNER(R10)  ; AND INDICATE SUBPROCESS
755          MOVL    PCB$L_EPID(R4),PCB$L_EOWNER(R10); AND ALSO COPY THE EXTENDED PID
756 110$:    MOVZWL  G^SCH$GL_MAXPIX,R7      ; INITIALIZE INDEX FOR NAME CHECK
757          TSTL    R6                      ; ANY LOGICAL NAME FOR NEW PROCESS?
758          BEQL    140$                    ; NO, SKIP SEARCH
759          INCL    R6                      ; Bump the length for the compare to
760                                          ; speed up the CMPC.
761 120$:    MOVL    @W^SCH$GL_PCBVEC[R7],R8 ; FETCH A PCB ADDRESS
762          CMPW    PCB$W_GRP(R8),PCB$W_GRP(R10)    ; SAME GROUP?
763          BNEQ    130$                    ; NO, TRY ANOTHER
764          CMPB    PCB$T_LNAME(R10),PCB$T_LNAME(R8)  ; AVOID CMPC3 MOST TIMES
765          BNEQ    130$                    ; NO MATCH, CONTINUE
766          CMPC3   R6,PCB$T_LNAME(R10),PCB$T_LNAME(R8)     ; COMPARE NAMES
767          BNEQ    130$                    ; NO MATCH, CONTINUE
768          MOVZWL  #SS$_DUPLNAM,R0         ; SET DUPLICATE NAME CODE
769          BRW     ABORT                   ; AND ABORT CREATION
770
771 126$:    BUG_CHECK KRPEMPTY,FATAL        ; OUT-OF-LINE KRP ALLOCATION BUGCHECK
772 130$:    SOBGTR  R7,120$                 ; CONTINUE SEARCH
773 140$:    MOVSTRING       LIM=255,-       ; MOV IMAGE NAME TO BUFFER
774                          SRC=IMAGE,-     ; FROM IMAGE STRING DESCRIPTOR
775                          DST=IMAGE       ; TO PQB$T_IMAGE
776          MOVSTRING       LIM=255,-       ; MOVE INPUT LOGICAL NAME TO BUFFER
777                          SRC=INPUT,-     ; FROM INPUT STRING DESCRIPTOR
778                          DST=INPUT       ; TO PQB$T_INPUT
779          MOVSTRING       LIM=255,-       ; MOVE OUTPUT LOGICAL NAME TO BUFFER
780                          SRC=OUTPUT,-    ; FROM OUTPUT STRING DESCRIPTOR
781                          DST=OUTPUT      ; TO PQB$T_OUTPUT
782          MOVSTRING       LIM=255,-       ; MOVE ERROR LOGICAL NAME TO BUFFER
783                          SRC=ERROR,-     ; FROM ERROR STRING DESCRIPTOR
784                          DST=ERROR       ; TO PQB$T_ERROR
785
786          MOVB    #^B1111,PQB$B_MSGMASK(R9) ; DEFAULT MESSAGE FLAGS = ALL OF THEM
787          BBC     #31,SP,1401$            ; IF SYSTEM SPACE STACK
788          BRW     145$                    ; THEN NO P1 SPACE DATA BASE AVAILABLE
789 1401$:   PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; SAVE MOVC REGISTERS
790
791 ;
792 ; TRANSLATE SYS$DISK USING G^LNM$SEARCH_ONE.
793 ; THIS TRANSLATES EXACTLY ONCE AND PASSES THE EQUIVALENCE STRING TO
794 ; THE CREATED PROCESS.  COMPLETELY BOGUS RESULTS WILL OCCUR IF THIS
795 ; TRANSLATION IS NOT EITHER A SYSTEM-WIDE LOGICAL NAME OR A PHYSICAL
796 ; DEVICE NAME.  THIS TOTALLY BOGUS IMPLEMENTATION MUST REMAIN THIS WAY
797 ; FOR COMPATIBILITY WITH PREVIOUS RELEASES.
798 ;
799 ; ALLOCATE A KERNEL REQUEST PACKET TO CONTAIN THE EQUIVALENCE STRING
800 ; (AND LOGICAL NAME WORK AREA) FOR G^LNM$SEARCH_ONE.
801
802          MOVAB   G^CTL$GL_KRPFL,R7       ; GET KRP LISTHEAD
803          REMQUE  @4(R7),R6               ; GET A KRP
804          BVS     126$                    ; BUG_CHECK IF ALL ARE IN USE
```

```
  805          PUSHL   R6                      ; SAVE POINTER TO KRP
  806          MOVZWL  G^EXE$GQ_SYSDISK,R0     ; GET DESCRIPTION OF SYS$DISK FOR CALL
  807          MOVL    G^EXE$GQ_SYSDISK+4,R1
  808          MOVZWL  LNM_TBL,R2              ; GET DESCRIPTION OF TABLE NAME
  809          MOVL    LNM_TBL+4,R3
  810          MOVZWL  #LNM_ATTR,R5            ; CASE_BLIND AND USER MODE
  811          JSB     G^LNM$SEARCH_ONE                ; TRANSLATE SYS$DISK (R4 HAS PCB)
  812          BLBS    R0,1405$                ; SUCCESS!
  813          CMPW    R0,#SS$_NOLOGNAM        ; NO TRANSLATION IS OKAY
  814          BNEQ    141$                    ; ELSE ABORT
  815          BRB     1406$                   ; GO RETURN KRP
  816 1405$:   MOVZBL  LNMX$T_XLATION(R6),R0   ; GET SIZE OF TRANSLATION
  817          MOVB    R0,PQB$T_DISK(R9)       ; STORE COUNT IN PQB
  818          MOVC3   R0,<LNMX$T_XLATION+1>(R6),-
  819                  <PQB$T_DISK+1>(R9)      ; COPY DEVICE NAME TO PQB
  820 1406$:   MOVL    #1,R0                   ; INDICATE SUCCESS
  821 141$:    MOVL    (SP)+,R6                ; RESTORE KRP POINTER
  822          INSQUE  (R6),@4(R7)             ; RETURN KRP
  823          BLBC    R0,146$                 ; ABORT IF ERROR
  824
  825 ; MOVE MINIMUM AND MAXIMUM AUTHORIZED SECURITY CLEARANCE RECORDS FROM
  826 ; CREATOR'S PHD INTO THE PQB. THE FOLLOWING ASSUME STATEMENTS GUARANTEE THAT
  827 ; WE CAN SAFELY PERFORM THIS WITH A SINGLE MOVC3 INSTRUCTION.
  828
  829          ASSUME PQB$S_MIN_CLASS EQ PHD$S_MIN_CLASS
  830          ASSUME PQB$S_MAX_CLASS EQ PHD$S_MAX_CLASS
  831          ASSUME PQB$R_MAX_CLASS EQ <PQB$R_MIN_CLASS + PQB$S_MIN_CLASS>
  832          ASSUME PHD$R_MAX_CLASS EQ <PHD$R_MIN_CLASS + PHD$S_MIN_CLASS>
  833
  834          MOVL    G^CTL$GL_PHD,R5         ; GET CREATOR'S PHD ADDRESS
  835          MOVC3   #<PHD$S_MIN_CLASS+PHD$S_MAX_CLASS>,-
  836                  PHD$R_MIN_CLASS(R5),-
  837                  PQB$R_MIN_CLASS(R9)
  838
  839          MOVC3   #PQB$S_DDSTRING,-
  840                  G^PIO$GT_DDSTRING,PQB$T_DDSTRING(R9) ; DEFAULT DIRECTORY
  841
  842 ; Move CLI and CLI table information from P1 space in one fell swoop:
  843 ;        G^CTL$GT_CLINAME     -> PQB$T_CLI_NAME
  844 ;        G^CTL$GT_TABLENAME   -> PQB$T_CLI_TABLE
  845 ;        G^CTL$GT_SPAWNCLI    -> PQB$T_SPAWN_CLI          Optional
  846 ;        G^CTL$GT_SPAWNTABLE -> PQB$T_SPAWN_TABLE                Optional
  847
  848          ASSUME PQB$T_CLI_TABLE   EQ <PQB$T_CLI_NAME  + PQB$S_CLI_NAME>
  849          ASSUME PQB$T_SPAWN_CLI   EQ <PQB$T_CLI_TABLE + PQB$S_CLI_TABLE>
  850          ASSUME PQB$T_SPAWN_TABLE EQ <PQB$T_SPAWN_CLI + PQB$S_SPAWN_CLI>
  851
  852          MOVZWL  #<PQB$S_CLI_NAME+-      ; ASSUME WE'LL MOVE ALL FIELDS
  853                    PQB$S_CLI_TABLE+-
  854                    PQB$S_SPAWN_CLI+-
  855                    PQB$S_SPAWN_TABLE>,-
  856                  R0
  857          BBS     #PRC$V_CLISPEC,STSFLG(AP),143$ ; PASSING ON CLI SPECIFICATION?
  858          MOVZWL  #<PQB$S_CLI_NAME+-      ; NO, ONLY MOVE FIRST TWO FIELDS
  859                    PQB$S_CLI_TABLE>,-
  860                  R0
  861 143$:    MOVC5   R0,-                    ; MOVE AS MANY FIELDS AS SPECIFIED
```

```
SYSCREPRC CREATE PROCESS SYSTEM SERVICE 10-MAY-1989 16:39:53 VAX MACRO V5.0-8 Page 17
X-32 EXE$CREPRC - CREATE PROCESS SYSTEM SERVI 19-SEP-1988 09:16:28 [SYS.SRC]SYSCREPRC.MAR;1

                        G^CTL$GT_CLINAME,-      ;  FROM P1 SPACE
                        #0,-                    ;   ZEROING OPTIONAL FIELDS AS NEEDED
                        #<PQB$S_CLI_NAME+-      ;    TO
                          PQB$S_CLI_TABLE+-     ;     THE
                          PQB$S_SPAWN_CLI+-     ;      FULL
                          PQB$S_SPAWN_TABLE>,-  ;       ALLOCATION
                        PQB$T_CLI_NAME(R9)      ;        IN THE PQB

        POPR    #^M<R0,R1,R2,R3,R4,R5>  ; RESTORE MOVC REGISTERS
        MOVB    G^CTL$GB_MSGMASK,PQB$B_MSGMASK(R9) ; USE CREATOR'S MESSAGE FLAGS
        MOVL    G^CTL$GL_UAF_FLAGS,PQB$L_UAF_FLAGS(R9) ; AND FLAGS FROM UAF RECORD
;
;       ESTABLISH STATUS FLAG SETTINGS FOR PROCESS
;
145$:   MOVL    STSFLG(AP),R8           ; GET STATUS FLAG ARGUMENT
        EXTZV   #STSFLGCNT,#<32-STSFLGCNT>,R8,R0        ; TEST MBZ FIELD
        BEQL    150$                    ; CORRECT IF ZERO
        MOVZWL  #SS$_IVSTSFLG,R0        ; ERROR, INVALID STATUS FLAG ARG
146$:   BRW     ABORT                   ; ABORT CREATION

150$:   MOVL    R8,PQB$L_CREPRC_FLAGS(R9) ; STORE FLAGS IN PQB
        CLRL    R7                      ; INITIALIZE INDEX FOR SCAN
160$:   FFS     R7,#STSFLGCNT,R8,R7     ; FIND AN ACTIVE STATUS FLAG
        BEQL    190$                    ; NONE, FINISHED WITH SCAN
        MOVAW   STSFLGTBL[R7],R1        ; POINT TO TRANSLATION ENTRY
        CVTBL   (R1)+,R0                ; GET PRIVILEGE BIT NUMBER  TO CHECK
        BGEQ    165$                    ; NEGATIVE MEANS POSSIBLY NOT PRIVILEGED
        CMPB    #-1,R0
        BEQL    170$                    ; NO PRIVILEGE REQUIRED
        TSTL    R11
        BEQL    170$                    ; NO PRIVILEGE REQUIRED FOR SUBPROCESS
        EXTZV   #0,#7,R0,R0             ; GET THE PRIVILEGE BIT NUMBER ONLY
165$:   BBS     R0,@PCB$L_PHD(R4),170$  ; THE CREATOR PROCESS MUST HAVE PRIVILEGE
        MOVZWL  #SS$_NOPRIV,R0          ; INDICATE NO PRIVILEGE ERROR
        BRW     ABORT                   ; AND ABORT PROCESS CREATION

170$:   CVTBL   (R1),R0                 ; GET BIT NUMBER IN STS
        BLSS    180$                    ; NOT NEEDED IN PCB
        BBSS    R0,PCB$L_STS(R10),180$  ; SET STSFLG IN NEW PCB
180$:   INCL    R7                      ; NEXT BIT
        BRB     160$                    ; CONTINUE SCAN
190$:                                   ; DONE WITH STSFLG
;
; MISC PQB FLAGS
;
        CLRW    PQB$W_FLAGS(R9)         ; INITIALIZE THE FLAGS
        BBC     #PRC$V_IMGDMP,STSFLG(AP),191$
        BISW    #PQB$M_IMGDMP,PQB$W_FLAGS(R9) ; REQUEST IMAGE DUMP
191$:   BBC     #PRC$V_DEBUG,STSFLG(AP),192$    ; If DEBUG bit set,
        BISW    #PQB$M_DEBUG,PQB$W_FLAGS(R9)    ;   then pass in PQB
192$:   BBC     #PRC$V_DBGTRU,STSFLG(AP),195$   ; If DBGTRU bit set,
        BISW    #PQB$M_DBGTRU,PQB$W_FLAGS(R9)   ;   then pass in PQB
195$:
;
; CHECK FOR MAXIMUM ALLOWED DETACHED PROCESSES
;
        TSTL    R11
```

```
  919          BEQL    200$                    ; NOT DETACHED
  920          PCB_IBN = <1@PCB$V_INTER>+<1@PCB$V_BATCH>+<1@PCB$V_NETWRK>
  921          BITL    #PCB_IBN,PCB$L_STS(R10)
  922          BNEQ    200$                    ; NO CHECK ON NETWORK, BATCH, OR INTERACTIVE
  923          MOVL    PCB$L_JIB(R10),R5       ; GET THE JIB OF NEW PROCESS
  924          MOVL    PCB$L_JIB(R4),R7        ; GET THE JIB OF PARENT PROCESS
  925          BNEQ    205$                    ; IF NO JIB - WE ARE BOOTING
  926 200$:    BRW     240$                    ; BRANCH AID
  927
  928          ASSUME JIB$W_MAXDETACH EQ JIB$W_MAXJOBS+2
  929
  930 205$:    MOVL    JIB$W_MAXJOBS(R7),-     ; PROPAGATE THE LIMITS
  931                  JIB$W_MAXJOBS(R5)
  932          BEQL    200$                    ; NO LIMIT
  933          CLRL    R6
  934          CLRL    R8
  935          PUSHR   #^M<R0,R1,R2,R3,R4,R9,R10,R11>
  936          MOVL    #2,R9                   ; PROCESS INDEX - AFTER NULL AND SWAPPER
  937
  938          ; Start of locked code
  939
  940          PMLREQ  END=225$                ; LOCK PAGES IN MEMORY TO LABEL 225$

      ;++
      ; NB: Co-routine address + 2 LWs have been placed on top of stack
      ;--
  941          LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
  942                  PRESERVE=NO             ; DON'T PRESERVE R0
  943 210$:    MOVL    @W^SCH$GL_PCBVEC[R9],R10 ; GET A PCB
  944          CMPL    R10,@W^SCH$GL_PCBVEC    ; IS IT NULL PROCESS PCB?
  945          BEQL    220$                    ; YES
  946          TSTL    PCB$L_OWNER(R10)        ; IS IT SUBPROCESS?
  947          BNEQ    220$
  948          BBS     #PCB$V_NETWRK,PCB$L_STS(R10),220$ ; DON'T COUNT NETWORK JOBS
  949          MOVL    PCB$L_JIB(R10),R11      ; JIB OF PROCESS BEING CHECKED
  950          CMPL    JIB$T_USERNAME(R7),JIB$T_USERNAME(R11) ; AVOID CMPC3 MOST TIMES
  951          BNEQ    220$                    ; NOT THE SAME USER
  952          CMPC3   #JIB$S_USERNAME,JIB$T_USERNAME(R7),JIB$T_USERNAME(R11)
  953          BNEQ    220$                    ; NOT THE SAME USER
  954          INCL    R8                      ; ONE MORE TOTAL JOB
  955          PCB_IB = <1@PCB$V_INTER>+<1@PCB$V_BATCH>
  956          BITL    #PCB_IB,PCB$L_STS(R10)
  957          BNEQ    220$                    ; INTERACTIVE OR BATCH
  958          INCL    R6                      ; ONE MORE TO COUNT AGAINST DETACHED
  959 220$:    AOBLEQ  G^SCH$GL_MAXPIX,R9,210$
  960          UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATABASE
  961                  NEWIPL=#IPL$_ASTDEL,-   ; LOWER IPL
  962                  PRESERVE=NO             ; DON'T PRESERVE R0
  963          PMLEND                          ; END OF LOCKED PAGES

      ;++
      ; NB: Co-routine address + 2 LWs have been removed from top of stack
      ;--
  964          POPR    #^M<R0,R1,R2,R3,R4,R9,R10,R11>
  965          MOVZWL  JIB$W_MAXJOBS(R7),R5
  966          BEQL    225$                    ; NO LIMIT
  967          SUBL    R5,R8
```

```
  968          BGEQU   230$                    ; OVER LIMIT (INCLUDING THIS PROCESS)
  969 225$:    CVTWL   JIB$W_MAXDETACH(R7),R5
  970          BEQL    240$                    ; NO LIMIT
  971          BLSS    230$                    ; NEGATIVE NUMBERS MEAN NONE ALLOWED
  972          SUBL    R5,R6
  973          BLSSU   240$                    ; OVER LIMIT (INCLUDING THIS PROCESS)
  974 230$:    MOVL    #SS$_EXPRCLM,R0
  975          BRW     ABORT
  976 ;
  977 240$:
  978 ;
  979          .DSABL  LSB
```

```
  981          .SBTTL  ESTABLISH QUOTAS FOR NEW PROCESS
  982 ;-----------------------------------------------------------------
  983 ;
  984 ;       PROCESS QUOTA BLOCK
  985 ;
  986 ;-----------------------------------------------------------------
  987
  988 ;
  989 ;       THE PROCESS QUOTA LIST, IF SUPPLIED, HAS THE FOLLOWING STRUCTURE
  990 ;       EACH QUOTA IS INTRODUCED AND IDENTIFIED BY A CODE BYTE, PQL$_?????,
  991 ;       WHICH IS FOLLOWED BY A LONGWORD CONTAINING THE QUOTA VALUE.
  992 ;
  993 ;       THE QUOTA VALUES SUPPLIED ARE MAXIMIZED WITH THE REQUIRED MINIMUM
  994 ;       VALUES AND REPLACE THE DEFAULT VALUE FOR EACH SPECIFIED QUOTA.
  995 ;       ONLY IF THE PROCESS CREATION IS SUCCESSFUL ARE THE DEDUCTIBLE
  996 ;       QUOTAS SUBTRACTED FROM THOSE OF A PROCESS CREATING A DETACHED PROCESS.
  997 ;
  998 ;       IF DUPLICATE QUOTA ENTRIES ARE FOUND, THE LAST ONE ENCOUNTERED IS
  999 ;       THE ONE THAT IS USED.
 1000 ;
 1001
 1002 QUOTALIST:                              ;
 1003         PUSHR   #^M<R0,R1,R2,R3,R4>     ; SAVE REGISTERS FOR MOVC
 1004         MOVC3   #<<PQL$_LENGTH-1>*4>,-  ; COPY DEFAULTS TO PQB
 1005                 G^PQL$AL_DEFAULT+4,-    ; AS ASSUMED VALUES FOR
 1006                 PQB$L_ASTLM(R9)         ; QUOTAS
 1007         POPR    #^M<R0,R1,R2,R3,R4>     ; RESTORE REGISTERS
 1008         MOVL    PCB$L_PHD(R4),R5        ; FOR SWAPPER POINT TO REAL PHD SINCE NO
 1009         CMPL    PCB$L_PID(R4),G^SCH$GL_SWPPID   ; IS IT THE SWAPPER
 1010         BEQL    5$                      ; YES, USE PCB POINTER TO PHD
 1011         MOVL    G^CTL$GL_PHD,R5         ; GET POINTER TO PHD WINDOW IN
 1012                                         ; CONTROL REGION FOR WINDOW
 1013 5$:     MOVL    QUOTA(AP),R7            ; GET POINTER TO QUOTA LIST
 1014         BEQL    NOQLIST                 ; NONE SPECIFIED
 1015 10$:    IFNORD  #1,(R7),30$             ; CHECK FOR ACCESSIBILITY
 1016         MOVZBL  (R7)+,R6                ; GET CODE
 1017         ASSUME  PQL$_LISTEND EQ 0       ;
 1018         BEQL    NOQLIST                 ; DONE IF PQL$_ENDLIST
 1019         IFNORD  #4,(R7),30$             ; CHECK QUOTA FOR ACCESSIBILITY
 1020         CMPW    R6,#PQL$_LENGTH         ; CHECK FOR LEGAL QUOTA NUMBER
 1021         BGEQ    20$                     ; INVALID IF GEQ
 1022         MOVL    (R7)+,PQB$L_ASTLM-4(R9)[R6] ; MERGE INTO PQB QUOTA LIST
 1023         BRB     10$                     ; GO GET NEXT QUOTA SPECIFIED
 1024 20$:    MOVZWL  #SS$_IVQUOTAL,R0        ; INVALID QUOTA LIST
 1025         BRW     ABORT                   ; SIGNAL ERROR CONDITION AND ABORT CREATE
 1026
 1027 30$:    BRW     ACCVIO                  ; ABORT WITH ACCESS VIOLATION
 1028
 1029 NOQLIST:                                ; DONE MERGING QUOTAS SPECIFIED
 1030         MOVAL   G^PQL$AL_MIN+<4*PQL$_LENGTH>,R6 ; SET POINTER TO BASE OF MIN VALUES
 1031         MOVAL   <PQB$L_ASTLM+<4*<PQL$_LENGTH-1>>>(R9),R3 ; SET BASE OF QUOTA VALUES
 1032         MOVZWL  #<PQL$_LENGTH-1>,R8     ; SET COUNT FOR SCAN
 1033         MOVL    PCB$L_JIB(R10),R7       ; GET ADDRESS OF JOB INFORMATION BLOCK
 1034         CLRL    R2                      ; INDICATE UNRESTRICTED QUOTAS ALLOWED
 1035         TSTL    R11                     ; IS IT A SUBPROCESS
 1036         BEQL    5$                      ; YES
 1037         IFPRIV  DETACH,10$              ; UNRESTRICTED IS OK
```

```
1038            IFPRIV  CMKRNL,10$
1039 5$:        MOVL    PCB$L_JIB(R4),R2        ; INDICATE RESTRICTED QUOTAS (JIB ADDRESS)
1040 10$:       CMPL    -(R3),-(R6)             ; CHECK AGAINST MINIMUM ALLOWABLE VALUE
1041            BGEQ    20$                     ; BR IF ABOVE MINIMUM
1042            MOVL    (R6),(R3)               ; FORCE TO MINIMUM
1043 20$:
1044            MOVL    (R3),R0                 ; GET QUOTA REQUEST VALUE
1045            BSBB    40$                     ; PROCESS QUOTA
1046            SOBGTR  R8,10$                  ; LOOP FOR ALL QUOTAS
1047            BRW     ITEMLIST                ; GO PROCESS THE ITEM LIST
1048
1049 40$:       CASE    R8,LIMIT=#1,<-          ; SWITCH ON TYPE OF QUOTA
1050                    QASTLM,-                ; 1 => AST LIMIT
1051                    QBIOLM,-                ; 2 => BUFFERED I/O LIMIT
1052                    QBYTLM,-                ; 3 => BUFIO BYTE COUNT LIMIT
1053                    QCPULM,-                ; 4 => CPU TIME LIMIT
1054                    QDIOLM,-                ; 5 => DIRECT I/O LIMIT
1055                    QFILLM,-                ; 6 => OPEN FILE LIMIT
1056                    QPGFLQUOTA,-            ; 7 => PAGING FILE QUOTA
1057                    QPRCLM,-                ; 8 => SUB-PROCESS LIMIT
1058                    QTQELM,-                ; 9 => TIMER QUEUE ENTRY LIMIT
1059                    QWSQUOTA,-              ;10 => WORKING SET QUOTA
1060                    QWSDEFAULT,-            ;11 => WORKING SET DEFAULT
1061                    QENQLM,-                ;12 => ENQUEUE LIMIT
1062                    QWSEXTENT,-             ;13 => WORKING SET EXTENT
1063                    QJTQUOTA-               ;14 => JOB-WIDE LOGICAL NAME TABLE QUOTA
1064                    >;
1065
1066 QJTQUOTA:                                  ; NO SPECIAL PROCESSING FOR JTQUOTA
1067            RSB                             ;
1068
1069 QASTLM:                                    ; AST LIMIT
1070            CMPW    R0,PHD$W_ASTLM(R5)      ; CHECK FOR IN LIMIT
1071            BLEQU   10$                     ; YES, CONTINUE
1072            TSTL    R2                      ; UNRESTRICTED DETACHED CREATE?
1073            BEQL    10$                     ; YES, ALLOW ANYTHING
1074            MOVW    PHD$W_ASTLM(R5),R0      ; NO, LIMIT TO MAXIMUM
1075 10$:       MOVW    R0,PCB$W_ASTCNT(R10)    ; SET AS WORKING AST COUNT
1076            MOVZWL  R0,(R3)                 ; Reflect ASTLM in the PQB
1077            RSB                             ; NEXT QUOTA
1078
1079 QBIOLM:                                    ; BUFFERED I/O LIMIT
1080            CMPW    R0,PCB$W_BIOLM(R4)      ; CHECK FOR IN LIMIT
1081            BLEQU   10$                     ; YES, CONTINUE
1082            TSTL    R2                      ; UNRESTRICTED DETACHED CREATE?
1083            BEQL    10$                     ; YES ALLOW ANYTHING
1084            MOVW    PCB$W_BIOLM(R4),R0      ; NO, LIMIT TO CURRENT VALUE
1085 10$:       MOVW    R0,PCB$W_BIOLM(R10)     ; SET LIMIT
1086            MOVW    R0,PCB$W_BIOCNT(R10)    ; AND WORKING COUNT
1087            RSB                             ; NEXT QUOTA
1088
1089 QBYTLM:                                    ; BUFFERED I/O BYTE LIMIT
1090            TSTL    R11                     ; DETACHED CREATE?
1091            BEQL    10$                     ; BR IF NOT
1092            TSTL    R2                      ; UNRESTRICTED QUOTAS?
1093            BEQL    5$                      ; YES
1094            CMPL    R0,JIB$L_ORG_BYTLM(R2)  ; IS IT WITHIN LIMITS
```

```
1095          BLEQU   5$                      ; YES
1096          MOVL    JIB$L_ORG_BYTLM(R2),R0  ; LIMIT TO CURRENT VALUE
1097 5$:      MOVL    R0,JIB$L_BYTLM(R7)      ; SET IN JIB OF MASTER PROCESS
1098          MOVL    R0,JIB$L_BYTCNT(R7)     ; BOTH LIMIT AND COUNT VALUES
1099          MOVL    R0,JIB$L_ORG_BYTLM(R7)  ; AND ORIGINAL VALUE FIELD
1100 10$:     RSB                             ; NEXT QUOTA
1101
1102 QCPULM:                                  ; CPU TIME LIMIT
1103          TSTL    R2                      ; UNRESTRICTED DETACHED CREATE?
1104          BEQL    20$                     ; YES, ALLOW ANYTHING
1105          TSTL    R11                     ; IS IT A DETACHED PROCESS
1106          BEQL    3$                      ; NO
1107          CMPL    R0,PHD$L_CPULIM(R5)     ; IS IT WITHIN LIMITS
1108          BLEQU   20$                     ; YES
1109          MOVL    PHD$L_CPULIM(R5),R0     ; LIMIT IT TO CURRENT VALUE
1110          BRB     20$
1111 3$:      MOVL    PHD$L_CPULIM(R5),R1     ; GET CURRENT PROCESS LIMIT
1112          BEQL    20$                     ; NO LIMIT ON CURRENT PROCESS
1113          SUBL    PHD$L_CPUTIM(R5),R1     ; DETERMINE REMAINING LIMIT
1114          BLEQU   10$                     ; BRANCH IF WILL EXCEED LIMIT
1115          TSTL    R0                      ; IS REQUESTED VALUE INFINITY?
1116          BNEQ    5$                      ; NO, GO SEE IF REQUEST CAN BE FILLED
1117          INCL    R1                      ; IF REQUESTING INFINITY, GIVE
1118          BICL2   #1,R1                   ;  HALF OF CALLER'S LIMIT BY CLEARING
1119          ROTL    #31,R1,R0               ;  LOW BIT AND ROTATING THAT TO SIGN BIT
1120          MOVL    R0,(R3)                 ; STORE NEW LIMIT IN PQB
1121 5$:      CMPL    R0,R1                   ; CHECK FOR IN LIMIT
1122          BLSSU   20$                     ; YES, CONTINUE
1123 10$:     BSBW    OVERCHECK               ; CHECK FOR LEGAL TO EXCEED
1124 20$:     RSB                             ; NEXT QUOTA
1125
1126 QDIOLM:                                  ; DIRECT I/O LIMIT
1127          CMPW    R0,PCB$W_DIOLM(R4)      ; CHECK FOR IN LIMIT
1128          BLEQU   10$                     ; YES, CONTINUE
1129          TSTL    R2                      ; UNRESTRICTED DETACHED CREATE?
1130          BEQL    10$                     ; YES, ALLOW ANYTHING
1131          MOVW    PCB$W_DIOLM(R4),R0      ; ELSE LIMIT TO CURRENT VALUE
1132 10$:     MOVW    R0,PCB$W_DIOLM(R10)     ; SET LIMIT
1133          MOVW    R0,PCB$W_DIOCNT(R10)    ; AND SET WORKING COUNT
1134          RSB                             ;
1135
1136 QFILLM:                                  ; OPEN FILE LIMIT
1137          TSTL    R11                     ; IS THIS A DETACHED CREATE?
1138          BEQL    10$                     ; BR IF NOT
1139          TSTL    R2                      ; UNRESTRICTED QUOTAS?
1140          BEQL    5$                      ; YES
1141          CMPW    R0,JIB$W_FILLM(R2)      ; IS IT WITHIN LIMITS
1142          BLEQU   5$                      ; YES
1143          MOVW    JIB$W_FILLM(R2),R0      ; LIMIT TO CURRENT VALUE
1144 5$:      MOVW    R0,JIB$W_FILCNT(R7)     ; SET FILE COUNT FOR MASTER PROCESS
1145          MOVW    R0,JIB$W_FILLM(R7)      ; AND LIMIT VALUE
1146 10$:     RSB                             ; NEXT QUOTA
1147
1148 QPGFLQUOTA:                              ; PAGE FILE QUOTA
1149          TSTL    R11                     ; DETACHED CREATE?
1150          BEQL    10$                     ; BR IF NOT
1151          TSTL    R2                      ; UNRESTRICTED QUOTAS?
```

```
1152          BEQL    5$                          ; YES
1153          CMPL    R0,JIB$L_PGFLQUOTA(R2)      ; IS IT WITHIN LIMITS
1154          BLEQU   5$                          ; YES
1155          MOVL    JIB$L_PGFLQUOTA(R2),R0      ; LIMIT TO CURRENT VALUE
1156 5$:      MOVL    R0,JIB$L_PGFLQUOTA(R7)      ; SET PAGE FILE QUOTA
1157          MOVL    R0,JIB$L_PGFLCNT(R7)        ; AND COUNT
1158 10$:     RSB                                 ; NEXT QUOTA
1159
1160 QPRCLM:                                      ; SUBPROCESS QUOTA
1161          TSTL    R11                         ; IS THIS A DETACHED CREATE?
1162          BEQL    10$                         ; BR IF NOT
1163          TSTL    R2                          ; UNRESTRICTED QUOTAS?
1164          BEQL    5$                          ; YES
1165          CMPW    R0,JIB$W_PRCLIM(R2)         ; IS IT WITHIN LIMITS
1166          BLEQU   5$                          ; YES
1167          MOVW    JIB$W_PRCLIM(R2),R0         ; LIMIT TO CURRENT VALUE
1168 5$:      MOVW    R0,JIB$W_PRCLIM(R7)         ; AND LIMIT
1169 10$:     RSB                                 ; NEXT QUOTA
1170
1171 QTQELM:                                      ; TIMER QUEUE ENTRY QUOTA
1172          TSTL    R11                         ; IS THIS A DETACHED CREATE?
1173          BEQL    10$                         ; BR IF NOT
1174          TSTL    R2                          ; UNRESTRICTED QUOTAS?
1175          BEQL    5$                          ; YES
1176          CMPW    R0,JIB$W_TQCNT(R2)          ; IS IT WITHIN LIMITS
1177          BLEQU   5$                          ; YES
1178          MOVW    JIB$W_TQCNT(R2),R0          ; LIMIT TO CURRENT VALUE
1179 5$:      MOVW    R0,JIB$W_TQCNT(R7)          ; SET TIMER QUEUE QUOTA FOR JOB
1180          MOVW    R0,JIB$W_TQLM(R7)           ; AND LIMIT VALUE
1181 10$:     RSB                                 ; NEXT QUOTA
1182
1183 QWSQUOTA:                                    ; WORKING SET QUOTA
1184          BSBB    MAXWSCNT                    ; LIMIT TO MAX WORKING SET COUNT
1185          ADDL    PHD$L_WSLIST(R5),R0         ; ADD BASE OF WORKING SET LIST
1186          CMPL    R0,PHD$L_WSQUOTA(R5)        ; CHECK FOR IN LIMIT
1187          BLSSU   10$                         ; YES,CONTINUE
1188          TSTL    R2                          ; UNRESTRICTED DETACHED PROCESS?
1189          BEQL    10$                         ; YES, USE AS IS
1190          ADDL3   #1,PHD$L_WSQUOTA(R5),R0     ; ELSE FORCE TO CORRECT MAXIMUM
1191 10$:     SUBL3   PHD$L_WSLIST(R5),R0,(R3)    ; REMOVE BIAS
1192          RSB                                 ; NEXT QUOTA
1193
1194 QWSEXTENT:                                   ; WORKING SET EXTENT
1195          BSBB    MAXWSCNT                    ; LIMIT TO MAX WORKING SET COUNT
1196          CMPL    R0,PQB$L_WSQUOTA(R9)        ; CHECK AGAINST QUOTA
1197          BGEQU   5$                          ; MORE, USE AS IS
1198          MOVL    PQB$L_WSQUOTA(R9),R0        ; MAXIMIZE WITH WSQUOTA
1199 5$:      ADDL    PHD$L_WSLIST(R5),R0         ; ADD BASE OF WORKING SET LIST
1200          CMPL    R0,PHD$L_WSEXTENT(R5)       ; CHECK FOR IN LIMIT
1201          BLSSU   10$                         ; YES,CONTINUE
1202          TSTL    R2                          ; UNRESTRICTED DETACHED PROCESS?
1203          BEQL    10$                         ; YES, USE AS IS
1204          ADDL3   #1,PHD$L_WSEXTENT(R5),R0    ; ELSE FORCE TO CORRECT MAXIMUM
1205 10$:     SUBL3   PHD$L_WSLIST(R5),R0,(R3)    ; REMOVE BIAS
1206          RSB                                 ; NEXT QUOTA
1207
1208 QWSDEFAULT:                                  ; WORKING SET LIST DEFAULT
```

```
1209          BSBB    MAXWSCNT                ; LIMIT TO MAXIMUM WORKING SET COUNT
1210          CMPL    R0,PQB$L_WSQUOTA(R9)    ; CHECK AGAINST QUOTA
1211          BLEQU   10$                     ; LESS, USE AS IS
1212          MOVL    PQB$L_WSQUOTA(R9),(R3)  ; MINIMIZE WITH WSQUOTA
1213 10$:     RSB                             ; NEXT QUOTA
1214
1215
1216 MAXWSCNT:                                ; LIMIT WSQUOTA OR DEFAULT TO MAXIMUL SIZE
1217          CMPL    R0,G^SGN$GL_MAXWSCNT    ; COMPARE WITH MAXIMUM WS LIST LENGTH
1218          BLEQU   10$                     ; BR IF WITHIN LEGAL RANGE
1219          MOVL    G^SGN$GL_MAXWSCNT,R0    ; FORCE TO LEGAL VALUE
1220 10$:     RSB                             ;
1221
1222 QENQLM:                                 ; ENQUEUE QUOTA
1223          TSTL    R11                     ; IS THIS A DETACHED CREATE?
1224          BEQL    10$                     ; BR IF NOT
1225          TSTL    R2                      ; UNRESTRICTED QUOTAS?
1226          BEQL    5$                      ; YES
1227          CMPW    R0,JIB$W_ENQCNT(R2)     ; IS IT WITHIN LIMITS
1228          BLEQU   5$                      ; YES
1229          MOVW    JIB$W_ENQCNT(R2),R0     ; LIMIT TO CURRENT VALUE
1230 5$:      MOVW    R0,JIB$W_ENQCNT(R7)     ; SET ENQUEUE QUOTA FOR JOB
1231          MOVW    R0,JIB$W_ENQLM(R7)      ; AND LIMIT VALUE
1232 10$:     RSB                             ; NEXT QUOTA
1233
```

```
1235          .SBTTL  OVERCHECK - CHECK FOR LEGAL TO EXCEED QUOTA
1236 ;++
1237 ; FUNCTIONAL DESCRIPTION:
1238 ;       OVERCHECK CHECKS TO SEE IF THE PROCESS BEING CREATED IS A DETATCHED
1239 ;       PROCESS.    IF A DETATCHED PROCESS IS BEING CREATED, CONTROL RETURNS
1240 ;       INLINE.  OTHERWISE THE CREATE IS ABORTED BY BRANCHING TO ABORT
1241 ;       WITH THE STATUS CODE SS$_NOQUOTA.
1242 ;
1243 ; INPUT PARAMETERS:
1244 ;       R4 - PCB ADDRESS OF CURRENT PROCESS
1245 ;       R5 - ADDRESS OF PHD FOR CURRENT PROCESS (WINDOW IN P1 SPACE)
1246 ;       R9 - ADDRESS OF PROCESS QUOTA BLOCK
1247 ;       R10- ADDRESS OF PCB FOR NEW PROCESS
1248 ;       R11- UIC FOR CREATED PROCESS (0 => SUBPROCESS)
1249 ;
1250 ;--
1251
1252 OVERCHECK:                                   ;
1253          TSTL    R11                         ; CHECK FOR SUBPROCESS CREATE
1254          BNEQ    10$                         ; YES, IGNORE OVER LIMIT
1255          MOVZWL  #SS$_EXQUOTA,R0             ; SET ERROR STATUS CODE
1256          BRW     ABORT                       ; NO, ABORT CREATE
1257
1258 10$:     RSB                                 ; RETURN
```

```
1261          .SBTTL  PROCESS THE ITEM LIST
1262 ;-----------------------------------------------------------------
1263 ;
1264 ;       PROCESS ITEM LIST
1265 ;
1266 ;-----------------------------------------------------------------
1267
1268 ;
1269 ;       THE PROCESS ITEM LIST, IF SUPPLIED, HAS THE FOLLOWING STRUCTURE
1270 ;       EACH ITEM HAS A 2 LONGWORD FIELD.  THE FIRST LONGWORD HAS TWO
1271 ;       SUBFIELDS, A WORD OF ITEM LENGTH, FOLLOWED BY A CODE WORD PRC$_?????,
1272 ;       WHICH IS FOLLOWED BY A LONGWORD CONTAINING THE ITEM VALUE.
1273 ;
1274 ;
1275 ;       IF DUPLICATE ITEM LIST ENTRIES ARE FOUND, THE LAST ONE ENCOUNTERED IS
1276 ;       THE ONE THAT IS USED.
1277 ;
1278 ;       NOTE THAT THE PAGE FILE INDEX AND CHARACTERISTICS WILL BE TREATED AS
1279 ;       ADVICE ONLY.  IF THEY ARE INVALID OR CANNOT BE MET (NO FILE AVAILABLE),
1280 ;       THE NORMAL ALGORITHM FOR ASSIGNING PAGE FILES WILL BE USED IN SHELL.
1281 ;
1282          ASSUME PRC$_PGFLCHAR EQ 1
1283          ASSUME PRC$_PGFLINDEX EQ 2
1284          ASSUME PRC$_INPUT_ATT EQ 3
1285          ASSUME PRC$_OUTPUT_ATT EQ 4
1286          ASSUME PRC$_ERROR_ATT EQ 5
1287 ;
1288 ITEMLIST:
1289          CMPL    (AP),#ITMLST_ARG        ; WAS THE ITEM LIST ARGUMENT SUPPLIED?
1290          BLSSU   NO_ITMLST               ; NO
1291          IFNORD  #4,ITMLST(AP),40$       ; CAN WE READ THE ITEMLIST POINTER
1292          MOVL    ITMLST(AP),R7           ; GET THE POINTER
1293          BEQL    NO_ITMLST               ; NONE SPECIFIED
1294 10$:     IFNORD  #4,(R7),40$             ; CAN WE READ THE CODE AND LENGTH
1295          MOVZWL  (R7)+,R0                ; GET LENGTH WORD
1296          MOVZWL  (R7)+,R6                ; GET THE CODE
1297          BEQL    NO_ITMLST               ; END OF THE LIST
1298          IFNORD  #4,(R7),40$             ; CAN WE READ THE ITEM VALUE
1299          BSBB    20$                     ; PROCESS THE ITEM
1300          ADDL    #4,R7
1301          BRB     10$                     ; NEXT
1302
1303 20$:     CASE    R6,LIMIT=#1,<-
1304                  ITM_PGFLCHAR,-
1305                  ITM_PGFLINDEX,-
1306                  ITM_INPUT_ATT,-
1307                  ITM_OUTPUT_ATT,-
1308                  ITM_ERROR_ATT,-
1309                  >
1310          MOVL    #SS$_BADPARAM,R0
1311          BRW     ABORT
1312
1313 40$:     BRW     ACCVIO
1314
1315 ITM_PGFLCHAR:                          ; PAGE FILE CHARACTERISTICS
1316          MOVW    (R7),PCB$W_PGFLCHAR(R10)
1317          RSB
```

```
1318
1319 ITM_PGFLINDEX:                          ; PAGE FILE INDEX
1320         MOVB    (R7),PCB$B_PGFLINDEX(R10)
1321         RSB
1322
1323 ITM_INPUT_ATT:                          ; SYS$INPUT ATTRIBUTES
1324         MOVL    (R7),PQB$L_INPUT_ATT(R9)
1325         RSB
1326
1327 ITM_OUTPUT_ATT:                         ; SYS$INPUT ATTRIBUTES
1328         MOVL    (R7),PQB$L_OUTPUT_ATT(R9)
1329         RSB
1330
1331 ITM_ERROR_ATT:                          ; SYS$INPUT ATTRIBUTES
1332         MOVL    (R7),PQB$L_ERROR_ATT(R9)
1333         RSB
1334
```

```
1336          .SBTTL  ACTIVATE NEW PROCESS
1337 ;
1338 ;       ACTIVATE NEW PROCESS
1339 ;
1340          .enabl  lsb
1341
1342 NO_ITMLST:
1343 ACTIVATE:                                   ;
1344          MOVL    R9,PCB$L_PQB(R10)          ; POINT NEW PCB TO PROCESS QUOTA BLOCK
1345          MOVL    R10,R4                     ; PCB ADDRESS OF NEW PROCESS
1346          MOVL    R4,(R10)                   ; BUILD QUEUE HEADER
1347          MOVL    R4,4(R10)                  ; FOR PCB
1348 10$:     PMLREQ  END=10001$                 ; LOCK PAGES IN MEMORY TO LABEL 10001$
      ;++
      ; NB: Co-routine address + 2 LWs have been placed on top of stack
      ;--
1349          LOCK    LOCKNAME=MMG,-             ; LOCK MMG DATABASE
1350                  SAVIPL=-(SP),-             ; SAVE CURRENT IPL
1351                  PRESERVE=NO                ; DON'T PRESERVE R0
1352          MOVL    CURPCB(FP),R6              ; GET PCB ADDRESS OF CREATOR
1353
1354          LOCK    LOCKNAME=SCHED             ; LOCK SCHED DATABASE
1355          MOVL    PCB$L_PHD(R6),R5           ; AND EXTRACT HEADER ADDRESS
1356 ;
1357 ; Look for a free PCB slot (i.e. one pointing to nullpcb).  Start at the slot after
1358 ; last PIX allocated and perform a round-robin scan.
1359 ;
1360          MOVZWL  G^SCH$GL_MAXPIX,R3         ; SAVE MAX PIX TO TEST WHEN TO WRAP AROUND
1361          SUBL3   S^#<SCH$C_SWPPIX+1>,R3,R1 ; LOOP COUNTER IS MAX LESS SWAPPER AND NUL
1362          MOVL    G^SCH$GL_PIXLAST,R7        ; SET INDEX FOR PIX SEARCH TO LAST ALLOCATED
1363          MOVL    G^SCH$AR_NULLPCB,R8        ; REFERENCE PCB ADDRESS (NULL PROCESS)
1364 20$:     INCL    R7                         ; MOVE TO THE NEXT PIX IN THE SEARCH
1365          CMPL    R7, R3                     ; IS THE PIX LARGER THAN THE MAXIMUM
1366          BLEQ    21$                        ; BRANCH IF R7 IS OK
1367          MOVL    S^#<SCH$C_SWPPIX+1>,R7     ; SET TO FIRST SLOT AFTER SWAPPER
1368 21$:     CMPL    R8,@W^SCH$GL_PCBVEC[R7]    ; FIND NON-ZERO PIX POINTING TO NULLPCB
1369          BEQL    30$                        ; GOT ONE, FREE SLOT
1370          SOBGTR  R1,20$                     ; OCCUPIED, TRY ANOTHER
1371 ;
1372 ; Error (max process count exceeded) - SCHED & MMG LOCKs still held.
1373 ;
1374 22$:     UNLOCK  LOCKNAME=SCHED             ; UNLOCK SCHED DATABASE
1375          BRW     60$                        ; NO FREE SLOTS AVAILABLE
1376
1377 ;
1378 ; Got a free slot, SCHED & MMG LOCKs still held.
1379 ;
1380 30$:     CMPW    G^SCH$GW_PROCLIM,G^SCH$GW_PROCCNT ; CHECK FOR MAX PROCESSES
1381          BLEQ    22$                        ; BR IF YES AND ABORT CREATE
1382 ;
1383 ; Update global data and create the internal and external process identifiers.
1384 ;
1385          MOVL    R7,G^SCH$GL_PIXLAST        ; SAVE NEW PIX AS LAST ALLOCATED PIX
1386          INCW    G^SCH$GW_PROCCNT           ; COUNT THIS PROCESS
1387          CMPW    G^SCH$GW_PROCCNT,-         ; Maintain maximum, concurrent
1388                  G^PMS$GL_PROCCNTMAX        ; process count
```

```
1389          BLEQU   31$
1390          MOVZWL  G^SCH$GW_PROCCNT,-
1391                  G^PMS$GL_PROCCNTMAX
1392 31$:
1393
1394 ; Note that the following code assumes that caller's of Create Process
1395 ; know what they are doing in the sense that there will never be more than
1396 ; one of the bits INTER, BATCH, and NETWRK set at the same time.
1397
1398          TSTL    R11
1399          BEQL    33$                     ; IT IS A SUBPROCESS - DON'T COUNT IT
1400          BBC     #PCB$V_INTER,PCB$L_STS(R10),32$
1401          INCW    G^SYS$GW_IJOBCNT                ; ONE MORE INTERACTIVE JOB
1402 32$:    BBC     #PCB$V_BATCH,PCB$L_STS(R10),33$
1403          INCW    G^SYS$GW_BJOBCNT                ; ONE MORE BATCH JOB
1404 33$:
1405 ;
1406 ; NOTE: The call to G^EXE$IPID_TO_EPID checks to make sure the IPID is valid,
1407 ;       therefore we must set the IPID in the new PCB and store the address
1408 ;       of the new PCB in the PCBVEC before we call G^EXE$IPID_TO_EPID.
1409 ;
1410          MOVL    R10,@W^SCH$GL_PCBVEC[R7]        ; SET POINTER TO PCB IN VECTOR OF PC
1411          MOVAW   @W^SCH$GL_SEQVEC[R7],R0 ; GET ADDRESS OF SEQUENCE NUMBER FOR SLOT
1412          INCW    (R0)                    ; NEXT SEQUENCE NUMBER FOR THIS PROCESS
1413          BGEQ    35$                     ; BR IF IN RANGE (POSITIVE PID)
1414          CLRW    (R0)                    ; ELSE, RESET SEQUENCE NUMBER
1415 35$:    MOVW    (R0),PCB$L_PID+2(R10)   ; SET SEQUENCE NUMBER
1416          MOVW    R7,PCB$L_PID(R10)       ; AND PIX TO FORM COMPLETE INTERNAL PID
1417          MOVL    PCB$L_PID(R10),R0       ; LOAD THE INTERNAL PID TO PASS TO ROUTINE
1418          JSB     G^EXE$IPID_TO_EPID      ; CONVERT IPID TO EPID, RETURN EPID IN R0
1419          MOVL    R0,PCB$L_EPID(R10)      ; STORE THE EXTENDED PID
1420          MOVL    R0,R7                   ; COPY EPID, SINCE WE CANNOT TOUCH THE OTHER
1421                                          ;   PCB AFTER WE SCHEDULE THE PROCESS
1422          TSTL    R11                     ; DETACHED CREATE?
1423          BEQL    38$                     ; BR IF NOT
1424          MOVL    PCB$L_JIB(R10),R0       ; GET JIB ADDRESS
1425          MOVL    PCB$L_PID(R10),JIB$L_MPID(R0)   ; AND SET ROOT PID FOR PROCESS TREE
1426 38$:    MOVZBL  #PRI$_TICOM,R2          ; SET PRIORITY INCREMENT CLASS
1427          JSB     G^SCH$CHSE              ; MAKE PROCESS EXECUTABLE, NON-RESIDENT
1428          TSTL    R11                     ; TEST FOR DETACHED CREATE
1429          BNEQ    40$                     ; BR IF CREATING DETACHED PROCESS
1430          INCW    PCB$W_PRCCNT(R6)        ; OTHERWISE ACCOUNT FOR SUB-PROCESS
1431          TSTL    PHD$L_CPULIM(R5)        ; CHECK FOR NO CPU LIMIT
1432          BEQL    40$                     ; NO LIMIT, DONT DEDUCT
1433          SUBL    PQB$L_CPULM(R9),PHD$L_CPULIM(R5); DEDUCT CPU TIME LIMIT
1434 40$:                                     ;
1435          UNLOCK  LOCKNAME=SCHED          ; UNLOCK SCHED DATABASE
1436          MOVL    16(SP),R0               ; GET EPID RETURN VALUE ADDRESS
1437          BEQL    55$                     ; NONE
1438          BLSS    70$                     ; SYSTEM SPACE ADDRESS
1439 45$:    UNLOCK  LOCKNAME=MMG,-          ; UNLOCK MMG DATABASE
1440                  NEWIPL=(SP)+            ; RESTORE PREVIOUS IPL
1441          MOVL    R7,(R0)                 ; RETURN EXTENDED PID FOR CREATED PROCESS
1442 50$:    PMLEND                          ; END OF LOCKED CODE

      ;++
      ; NB: Co-routine address + 2 LWs have been removed from top of stack
```

```
                ;--
1443                    MOVZWL  #SS$_NORMAL,R0          ; SET NORMAL COMPLETION STATUS
1444                    RET                             ; AND RETURN TO CALLER
1445
1446 55$:              UNLOCK  LOCKNAME=MMG,-          ; UNLOCK MMG DATABASE
1447                            NEWIPL=(SP)+,-          ; RESTORE PREVIOUS IPL
1448                            PRESERVE=NO             ; DON'T PRESERVE R0
1449                    BRB     50$                     ; AND EXIT
1450
1451 60$:              MOVZWL  #SS$_NOSLOT,R0          ; SET ERROR CODE FOR NO SLOT AVAILABLE
1452 65$:              UNLOCK  LOCKNAME=MMG,-          ; UNLOCK MMG DATABASE
1453                            NEWIPL=(SP)+            ; RESTORE PREVIOUS IPL
1454                    PMLEND                          ; END OF LOCKED CODE

                ;++
                ; NB: Co-routine address + 2 LWs have been removed from top of stack
                ;--
1455                    BRB     ABORT                   ; ABORT CREATION, EXCEEDED QUOTA
1456
1457 70$:              CMPL    R0,G^MMG$GL_NPAGEDYN    ; CHECK FOR IN NONPAGED POOL
1458                    BLSSU   45$                     ; NO, DROP IPL TO STORE EPID
1459                    CMPL    R0,G^EXE$GL_INTSTK      ; HIGH LIMIT
1460                    BGEQU   45$                     ; OUT OF POOL OR INTERRUPT STACK
1461                    MOVL    R7,(R0)                 ; STORE EXTENDED PID FOR CREATED PROCESS
1462                    UNLOCK  LOCKNAME=MMG,-          ; UNLOCK MMG DATABASE
1463                            NEWIPL=(SP)+,-          ; RESTORE PREVIOUS IPL
1464                            PRESERVE=NO             ; DON'T PRESERVE R0
1465                    BRB     50$                     ; AND RETURN SUCCESS
1466 10001$:                                           ; MARKER FOR END OF NONPAGABLE AREA
1467                    .dsabl lsb
```

```
1469          .SBTTL  ABORT PROCESS CREATION
1470 ;
1471 ;       ABORT PROCESS CREATION AFTER DETECTING ANY ERROR
1472 ;
1473
1474
1475 ABRT2:                                  ; ABORT WITH PCB ONLY
1476          CLRL    R9                     ; INDICATE NO PQB ALLOCATED
1477 ABORT:                                  ; ABORT WITH BOTH PCB AND PQB ALLOCATED
1478          MOVL    R0,R7                  ; SAVE STATUS CODE
1479          TSTL    R11                    ; IS THIS A DETACHED PROCESS CREATE?
1480          BEQL    10$                    ; BR IF NOT
1481          MOVL    PCB$L_JIB(R10),R0      ; GET JIB ADDRESS FOR RELEASE
1482          BEQL    10$                    ; BR IF NONE
1483          BSBW    EXE_DEANONPAGED        ; DEALLOCATE JIB
1484          BRB     20$                    ;
1485 10$:     MOVL    CURPCB(FP),R4          ; GET PCB ADDRESS FOR SURE
1486          MOVL    PCB$L_JIB(R4),R2       ; FETCH JIB ADDRESS
1487          PMLREQ  END=20$                ; Lock down pages

      ;++
      ; NB: Co-routine address + 2 LWs have been placed on top of stack
      ;--
1488          LOCK    LOCKNAME=MMG,-         ; Get MMG spinlock
1489                  SAVIPL=-(SP),-
1490                  PRESERVE=NO
1491          ADDL    #SWP$C_SHELLPFIL,JIB$L_PGFLCNT(R2); RETURN PAGE FILE QUOTA
1492                                         ; FOR SHELL PAGES
1493          UNLOCK  LOCKNAME=MMG,-         ; Unlock MMG
1494                  NEWIPL=(SP)+,-
1495                  PRESERVE=NO
1496          PMLEND                         ; Through with locked pages

      ;++
      ; NB: Co-routine address + 2 LWs have been removed from top of stack
      ;--
1497          ADAWI   #-1,JIB$W_PRCCNT(R2)   ; CORRECT SUBPROCESS COUNT
1498
1499 20$:     MOVL    PCB$Q_PRIV+ARB$L_RIGHTSLIST+8(R10),R0   ; GET EXTENDED RIGHTS LIS
1500          BEQL    25$                    ; BRANCH IF NONE
1501          BSBB    EXE_DEANONPAGED        ; DEALLOCATE RIGHTS LIST
1502 25$:     MOVL    R10,R0                 ; ADDRESS OF NEW PCB
1503          BSBB    EXE_DEANONPAGED        ; DEALLOCATE QUOTA BUFFER
1504          TSTL    R9                     ; ADDRESS OF PQB IF ANY
1505          BEQL    30$                    ; BR IF NONE ALLOCATED
1506          $INSQTI (R9),G^EXE$GQ_PQBIQ    ; DEALLOCATE PQB TO LOOKASIDE LIST
1507                                         ; NOTE: R0 is scratch in $INSQTI macro
1508 30$:     MOVL    R7,R0                  ; RESTORE STATUS CODE
1509          RET                            ; AND RETURN TO CALLER
1510
1511 EXE_DEANONPAGED:
1512          JMP     G^EXE$DEANONPAGED      ; BRANCH AID TO REACH THIS ROUTINE
```

```
1514          .SBTTL  MOVSTR - STRING COPY SUBROUTINE
1515 ;+
1516 ; FUNCTIONAL DESCRIPTION:
1517 ;       MOVSTR VALIDATES AND COPIES A STRING FROM THE ARGUMENT LIST
1518 ;       TO THE PROCESS QUOTA BUFFER.  IF ANY ERROR IS DETECTED, THE
1519 ;       SERVICE CALL IS EXITED VIA A RET INSTRUCTION WITH R0 CONTAINING
1520 ;       THE ERROR STATUS CODE.
1521 ;
1522 ; CALLING SEQUENCE:
1523 ;       BSB     MOVSTR
1524 ;       .BYTE   <MAXIMUM_STRING_LENGTH>
1525 ;       .BYTE   <AP_OFFSET_TO_SOURCE_DESCRIPTOR>
1526 ;       .BYTE   <PQB_OFFSET_OF_DESTINATION>
1527 ;
1528 ; INPUT PARAMETERS:
1529 ;       R9 - PQB BASE ADDRESS
1530 ;       @(SP) - MAXIMUM STRING LENGTH
1531 ;       @(SP)+1 - AP OFFSET TO SOURCE STRING DESCRIPTOR
1532 ;       @(SP)+2 - PQB OFFSET FOR DESTINATION COUNTED STRING
1533 ;
1534 ; OUTPUT PARAMETERS:
1535 ;       SPECIFIED AREA IN PQB RECEIVES SOURCE STRING
1536 ;       R2 - POINTS TO INPUT DESCRIPTOR IF NULL OUTPUT ADDRESS
1537 ;
1538 ; COMPLETION CODES:
1539 ;       SS$_ACCVIO  -  ACCESS VIOLATION FETCHING DESCRIPTOR OR STRING
1540 ;       SS$_IVLOGNAM - INVALID LOGICAL NAME (COUNT OUT OF RANGE)
1541 ;
1542 ;-
1543
1544 MOVSTR:                                 ; MOVE STRING TO PQB
1545          MOVL    (SP),R5                ; GET BASE OF PARAMETERS
1546          ADDL    #4,(SP)                ; INCREMENT RETURN ADDRESS
1547          PUSHR   #^M<R4,R5,R6,R7,R8>    ; SAVE REGISTERS
1548          MOVZBL  (R5)+,R8               ; GET STRING LIMIT
1549          MOVZBL  (R5)+,R3               ; GET ARGLIST OFFSET
1550          MOVL    (AP)[R3],R2            ; FETCH DESCRIPTOR ADDRESS
1551          BEQL    MOVEXIT                ; NONE, EXIT
1552          IFNORD  #8,(R2),ACCVIO         ; MUST BE ABLE TO READ DESCRIPTOR
1553          MOVQ    (R2),R6                ; FETCH DESCRIPTOR
1554          TSTW    R6                     ; CHECK FOR NULL COUNT
1555          BEQL    MOVEXIT                ; YES, NULL STRING
1556          CMPW    R6,R8                  ; CHECK UPPER LIMIT ON STRING
1557          BLEQU   10$                    ; BR IF WITHIN LIMIT
1558          MOVZWL  #SS$_IVLOGNAM,R0       ; SET ERROR CODE
1559          BRW     ABORT                  ; AND ABORT CREATE
1560
1561 10$:     IFNORD  R6,(R7),ACCVIO         ; CHECK ACCESSIBILITY
1562          MOVZWL  (R5),R3                ; GET PQB OFFSET
1563          BEQL    MOVEXIT                ; JUST CHECK PROTECTION
1564          MOVAB   (R9)[R3],R3            ; COMPUTE ADDRESS IN PQB
1565          MOVB    R6,(R3)+               ; SET COUNT FOR STRING
1566          MOVC3   R6,(R7),(R3)           ; COPY STRING TO BUFFER
1567 MOVEXIT:                                ;
1568          POPR    #^M<R4,R5,R6,R7,R8>    ; RESTORE REGISTERS
1569          RSB                            ; AND RETURN
1570
```

```
 1571 ACCVIO: MOVZWL  #SS$_ACCVIO,R0          ; SET ERROR CODE
 1572         BRW     ABORT                   ;
```

```
1574          .SUBTITLE       ALLOCPQB - Allocate PQB from paged pool
1575 ;+
1576 ; Functional Description:
1577 ;
1578 ;       This subroutine merely allocates a PQB from paged pool. If the
1579 ;       allocation fails, then we have to abort.
1580 ;
1581 ; Input Parameters:
1582 ;
1583 ;       R4  - Address of PCB of creating process
1584 ;       R10 - Address of chunk of nonpaged pool that will become PCB
1585 ;               of new process.
1586 ;
1587 ; Implicit Input:
1588 ;
1589 ;       Running at IPL$_ASTDEL as a result of successful PCB allocation
1590 ;
1591 ; Ouptut Parameters:
1592 ;
1593 ;       If allocation is successful
1594 ;
1595 ;               R2 - Address of PQB
1596 ;
1597 ;       If the allocation fails and the creator waits for resources,
1598 ;
1599 ;               the PCB pointed to by R10 is deallocated and the process
1600 ;               is put into a resource wait state. When paged pool becomes
1601 ;               available, the process resumes execution at the beginning
1602 ;               of the service.
1603 ;
1604 ;       If the allocation fails and resource wait is disabled.
1605 ;
1606 ;               a simple failure status (SS$_INSFMEM) is returned.
1607 ;
1608 ; Side Effects:
1609 ;
1610 ;       R0 through R3 are modified
1611 ;-
1612
1613 ALLOCPQB:                               ; Allocate process quota block
1614          MOVZWL  #PQB$C_LENGTH,R1       ; Set structure size
1615          PUSHL   R1                     ; Save request size
1616          JSB     G^EXE$ALOPAGED         ; Attempt to allocate packet
1617          POPL    R1                     ; Restore request size
1618          BLBC    R0,B^DEALLOCATE_PCB    ; If low bit clear, no packet allocated
1619          MOVW    R1,PQB$W_SIZE(R2)      ; Insert size of allocated block
1620          MOVZBW  #DYN$C_PQB,-           ; Insert data structure type
1621                  PQB$B_TYPE(R2)         ;  and clear adjacent byte
1622 10$:     RSB                            ;  and return
1623
1624 DEALLOCATE_PCB:
1625          MOVL    R10,R0                 ; Get new PCB address
1626          BSBW    EXE_DEANONPAGED        ; Give it back
1627          MOVZWL  #SS$_INSFMEM,R0        ; Return with error
1628          RET
1629
1630          .END
```

14 PROCSTRT.LIS

```
PROCSTRT - PROCESS STARTUP AND INITIALIZATION 10-MAY-1989 16:39:04 VAX MACRO V5.0-8 Page 0
Table of contents
```

```
PROCSTRT - PROCESS STARTUP AND INITIALIZATION 10-MAY-1989 16:39:04 VAX MACRO V5.0-8 Page 1
X-23 28-OCT-1988 08:51:22 [SYS.SRC]PROCSTRT.MAR;1 (1)

    1          .TITLE  PROCSTRT - PROCESS STARTUP AND INITIALIZATION
    2          .IDENT  'X-23'
    3
    4 ;
    5 ;*****************************************************************************
    6 ;*                                                                           *
    7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984, 1987 BY                            *
    8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
    9 ;*  ALL RIGHTS RESERVED.                                                     *
   10 ;*                                                                           *
   11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
   12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
   13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
   14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
   15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
   16 ;*  TRANSFERRED.                                                             *
   17 ;*                                                                           *
   18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
   19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
   20 ;*  CORPORATION.                                                             *
   21 ;*                                                                           *
   22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
   23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
   24 ;*                                                                           *
   25 ;*                                                                           *
   26 ;*****************************************************************************
   27
   28 ;++
   29 ; FACILITY: EXECUTIVE, PROCESS CREATION SYSTEM SERVICE
   30 ;
   31 ; ABSTRACT:
   32 ;       PROCSTRT CONTAINS THE CODE NECESSARY TO CONCLUDE THE CREATION
   33 ;       OF A PROCESS WHICH MUST BE EXECUTED IN THE CONTEXT OF THAT PROCESS.
   34 ;
   35 ; ENVIRONMENT:
   36 ;       MODE=KERNEL, EXECUTING IN CONTEXT OF NEW PROCESS
   37 ;
   38 ; AUTHOR: R. I. HUSTVEDT        , CREATION DATE: 27-DEC-76
   39 ;
   40 ; MODIFIED BY:
   41 ;
   42 ;       X-23    CWH5123         CW Hobbs                28-Oct-1988
   43 ;               Use final names for CWPS/PSCAN queue headers.
   44 ;
   45 ;       X-22    CWH5122         CW Hobbs                29-Aug-1988
   46 ;               Initialize the CWPS and PSCAN queues.
   47 ;
   48 ;       X-21    JEJ0529         James E Johnson         28-Jun-1988
   49 ;               Initialize the CTL$GQ_RMCB_QUE.
   50 ;
   51 ;       X-20    WMC0020         Wayne Cardoza           05-Jan-1988
   52 ;               Move the IMGRESET call to avoid lowering IPL before the PQB is
   53 ;               deallocated.
   54 ;
   55 ;       X-19    HH0300          Hai Huang               09-Oct-1987
   56 ;               Add VVIEF support.
   57 ;
```

```
;       X-18    JWT0298         Jim Teague              27-Aug-1987
;               Pass PQB DEBUG and DBGTRU flags through as CLI
;               flags.  Formerly, PROCSTRT always cleared the CLI
;               flags.
;
;       X-17    SSA0002         Stan Amway              25-Aug-1987
;               Limit WSQUOTA to 64K.
;
;       X-16    SF00016         Stephen Fiorelli        15-Jul-1987
;               Large working set support.
;
;       X-15    WMC0015         Wayne Cardoza           01-Jul-1987
;               Allow more message vectors.
;
;       X-14    RNG5014         Rod Gamache             12-Mar-1987
;               Add some performance optimizations - get PCB from CTL$GL_PCB.
;
;       X-13    WCT0032         Ward C. Travis           2-Mar-1987
;               Update remaining old lookaside  listhead references
;               to reflect that they are now interlocked queues.
;
;       X-12    LJK4015         Lawrence J. Kenah       28-Jan-1987
;               Map the XQP only once during the life of a process. The
;               change is simply moving the call to XQPMERGE before
;               the label EXE$PROCIMGACT.
;
;       X-11    DDP0033         Derrell D. Piper        27-Jan-1987
;               Add initialization for JPI context segment queue
;               header (CTL$GQ_JPICTX).
;
;       X-10    WMC0003         Wayne Cardoza           26-Jan-1987
;               No CMKRNL needed for the P1 CRETVA.
;
;       X-9     SSA0001         Stan Amway              16-Oct-1986
;               Support for exec mode rundown routines.
;
;       X-8     WMC0002         Wayne Cardoza           29-Jul-1986
;               Get rid of re-executeable flag.
;
;       X-7     SJF             Stu Farnham             30-Jun-1986
;               Resolve conflicts from merge of SMP into mainline.
;
;       X-5     WMC0001         Wayne  Cardoza          18-Jun-1986
;               Initialization routine should return status.
;
;       X-4     SF04002         Stephen Fiorelli        14-May-1986
;               Add a level of indirection when referencing exe$exceptable.
;               Pointer cell to this table lives in the base image.
;
;       X-3     SF04001         Stephen Fiorelli        07-Apr-1986
;               Resolve conflicts introduced from merge of exec_reorg_b17
;               thread into the mainline.
;
;       X-2     ACG0505         Andrew C. Goldstein,    6-Nov-1985  18:22
;               Deny control access to group and world in log name protection
;
;       X-1C8   TCM0004         Trudy C. Matthews       20-Nov-1985
```

```
                        Another error in initialization routine: JOB_TABLE's PARENT
                        and QUOTA fields are not being addressed properly.
;
;       X-1C7   TCM0003         Trudy C. Matthews       19-Nov-1985
;               Fix addressing errors in new initialization routine.
;
;       X-1C6   TCM0002         Trudy C. Matthews       18-Nov-1985
;               The LNM_SYSTEM_DIR_LNMTH structure is no longer directly
;               accessible; instead access it through a pointer called
;               LNM_AR_SYSTEM_DIR_LNMTH.  This involves changing some code
;               references and adding an initialization routine,
;               INI$PROCSTRT_INIT, that copies the address of the structure
;               into local data structures GROUP_TABLE and JOB_TABLE.
;
;
;       X-1C5   TCM0001         Trudy C. Matthews       4-Nov-1985
;               Add ALIGNMENT parameter to DECLARE_PSECT macro.
;
;       V04-002 RAS0332         Ron Schaefer            14-Sep-1984
;               Check for RMS$_BUSY status in the RMS exit handler
;               so as to prevent an infinite loop if the handler
;               has interrupted RMS rundown badly.   In that case,
;               give up on trying to do rundown cleanly.
;               Also, change the rundown type to do a full PPF rundown.
;
;       V04-001 JWT0195         Jim Teague              11-Sep-1984
;               Replace RMS exec mode exit handler for proper rundown
;               of single-image processes.
;
;--
```

```
146          .SBTTL  DECLARATIONS
147 ;
148 ; INCLUDE FILES:
149 ;
150
151          $BOOSTATEDEF                    ; DEFINE BOOTSTRAP STATE FLAGS
152          $CCBDEF                         ; CHANNEL CONTROL BLOCK DEFINITIONS
153          $CHFDEF                         ; CONDITION HANDLER DEFINITIONS
154          $CLIDEF                         ; CLI definitions
155          $CLIMSGDEF                      ; COMMAND INTERPRETER STATUS CODES
156          $CPUDEF                         ; DEFINE PER-CPU DATA BLOCK OFFSETS
157          $DYNDEF                         ; DYNAMIC STRUCTURE TYPE CODES
158          $IACDEF                         ; IMAGE ACTIVATION FLAGS
159          $IHDDEF                         ; IMAGE HEADER DESCRIPTOR DEFINITIONS
160          $IMGACTDEF                      ; IMAGE ACTIVATOR ARGUMENTS
161          $JIBDEF                         ; DEFINE JIB OFFSETS
162          $JPIDEF                         ; JPI ITEM CODES
163          $IMPDEF                         ; RMS IMPURE AREA DEFINITIONS
164          $LNMDEF                         ; LOGICAL NAME DEFINITIONS
165          $LNMSTRDEF                      ; LOGICAL NAME STRUCTURE DEFINITIONS
166          $OPDEF                          ; SYMBOLIC NAMES FOR INSTRUCTION OPCODES
167          $ORBDEF                         ; DEFINE OBJECT RIGHTS BLOCK OFFSETS
168          $PCBDEF                         ; DEFINE PCB OFFSETS
169          $PHDDEF                         ; DEFINE PROCESS HEADER
170          $PQBDEF                         ; DEFINE PROCESS QUOTA BLOCK OFFSETS
171          $PRDEF                          ; DEFINE PROCESSOR REGISTERS
172          $PRTDEF                         ; DEFINE PAGE PROTECTION VALUES
173          $PRVDEF                         ; PRIVILEGE BIT DEFINITIONS
174          $PSLDEF                         ; DEFINE PSL FIELDS
175          $RMSDEF                         ; DEFINE RMS ERROR STATUSES
176          $SECDEF                         ; SECTION FLAGS
177          $SGNDEF                         ; DEFINE SYSGEN CONSTANTS
178          $SSDEF                          ; DEFINE SYSTEM STATUS CODES
179          $STSDEF                         ; DEFINE STATUS CODE FIELDS
180
181 ;
182 ; ASSUMPTIONS ABOUT THE STRUCTURE OF LOGICAL NAME AND OBJECT RIGHTS BLOCKS:
183 ;
184
185          ASSUME  LNMB$L_FLINK,       EQ,  0
186          ASSUME  LNMB$L_FLINK+4,     EQ,  LNMB$L_BLINK
187          ASSUME  LNMB$L_BLINK+4,     EQ,  LNMB$W_SIZE
188          ASSUME  LNMB$W_SIZE+2,      EQ,  LNMB$B_TYPE
189          ASSUME  LNMB$B_TYPE+1,      EQ,  LNMB$B_ACMODE
190          ASSUME  LNMB$B_ACMODE+1,    EQ,  LNMB$L_TABLE
191          ASSUME  LNMB$L_TABLE+4,     EQ,  LNMB$B_FLAGS
192          ASSUME  LNMB$B_FLAGS+1,     EQ,  LNMB$T_NAME
193
194          ASSUME  LNMX$B_FLAGS,       EQ,  0
195          ASSUME  LNMX$B_FLAGS+1,     EQ,  LNMX$B_INDEX
196          ASSUME  LNMX$B_INDEX+1,     EQ,  LNMX$W_HASH
197          ASSUME  LNMX$W_HASH+2,      EQ,  LNMX$T_XLATION
198
199          ASSUME  LNMTH$B_FLAGS,      EQ,  0
200          ASSUME  LNMTH$B_FLAGS+1,    EQ,  LNMTH$L_HASH
201          ASSUME  LNMTH$L_HASH+4,     EQ,  LNMTH$L_ORB
202          ASSUME  LNMTH$L_ORB+4,      EQ,  LNMTH$L_NAME
```

```
        ASSUME  LNMTH$L_NAME+4,     EQ,  LNMTH$L_PARENT
        ASSUME  LNMTH$L_PARENT+4,   EQ,  LNMTH$L_CHILD
        ASSUME  LNMTH$L_CHILD+4,    EQ,  LNMTH$L_SIBLING
        ASSUME  LNMTH$L_SIBLING+4,  EQ,  LNMTH$L_QTABLE
        ASSUME  LNMTH$L_QTABLE+4,   EQ,  LNMTH$L_BYTESLM
        ASSUME  LNMTH$L_BYTESLM+4,  EQ,  LNMTH$L_BYTES

        ASSUME  ORB$L_OWNER,        EQ,  0
        ASSUME  ORB$L_OWNER+4,      EQ,  ORB$L_ACL_MUTEX
        ASSUME  ORB$L_ACL_MUTEX+4,  EQ,  ORB$W_SIZE
        ASSUME  ORB$W_SIZE+2,       EQ,  ORB$B_TYPE
        ASSUME  ORB$B_TYPE+1,       EQ,  ORB$B_FLAGS
        ASSUME  ORB$B_FLAGS+3,      EQ,  ORB$W_REFCOUNT
        ASSUME  ORB$W_REFCOUNT+2,   EQ,  ORB$Q_MODE_PROT
        ASSUME  ORB$Q_MODE_PROT+8,  EQ,  ORB$L_SYS_PROT
        ASSUME  ORB$L_SYS_PROT+4,   EQ,  ORB$L_OWN_PROT
        ASSUME  ORB$L_OWN_PROT+4,   EQ,  ORB$L_GRP_PROT
        ASSUME  ORB$L_GRP_PROT+4,   EQ,  ORB$L_WOR_PROT
        ASSUME  ORB$L_WOR_PROT+4,   EQ,  ORB$L_ACL_COUNT
        ASSUME  ORB$L_ACL_COUNT+4,  EQ,  ORB$L_ACL_DESC
        ASSUME  ORB$L_ACL_DESC+4,   EQ,  ORB$R_MIN_CLASS
        ASSUME  ORB$R_MIN_CLASS+ORB$S_MIN_CLASS,-
                                    EQ,  ORB$R_MAX_CLASS
        ASSUME  ORB$R_MAX_CLASS+ORB$S_MAX_CLASS,-
                                    EQ,  ORB$K_LENGTH

;
; MACROS:
;

        .MACRO  CRELNM,XLATION,XLATION_ATTR,LNMX,LNMB
        BSBW    CRELNM
        .WORD   <XLATION>
        .WORD   <XLATION_ATTR>
        .WORD   <LNMX>
        .WORD   <LNMB>
        .END    CRELNM

;
; EQUATED SYMBOLS:
;

NXTKVEC=0                       ;OFFSET TO NEXT FREE KERNEL VECTOR
NXTEVEC=256                     ;OFFSET TO NEXT FREE EXEC VECTOR
NXTRVEC=512                     ;OFFSET TO NEXT FREE RUNDWN VECTOR
NXTERVEC=768                    ;OFFSET TO NEXT FREE EXEC MODE RUNDWN VECTOR
NXTMVEC=1024                    ;OFFSET TO NEXT MESSAGE VECTOR
```

```
251
252 ;
253 ; OWN STORAGE:
254 ;
255
256         DECLARE_PSECT   EXEC$PAGED_DATA,ALIGNMENT=PAGE  ; PAGED PSECT
257
258         UNIVERSAL_SYMBOL        EXE$GQ_SYSDISK
259 ;EXE$GQ_SYSDISK::                        ; DESCRIPTOR FOR SYS$DISK
260         .ASCID  /SYS$DISK/
261 DEFDESC:                                ; DEFAULT IMAGE FILE NAME
262         .ASCID  /.EXE/
263
264 CHARS:  .ASCII  /0123456789ABCDEF/      ; CHARS FOR OCTAL (HEX) -> ASCII CONVS
265
266 ;
267 ; CATCH ALL HANDLER FATAL CONDITION MESSAGE SUFFIX.
268 ;
269
270 SUFFIX: .ASCIZ  /image exit forced./    ;
271
272 ;
273 ; STRINGS FOR IMAGE DUMP MERGE.
274 ;
275
276 DEFAULTNAMDSC:
277         .ASCID  /SYS$LIBRARY:.EXE/
278 IMGDMPNAM:
279         .ASCID  /IMGDMP/
280
281 ;
282 ; TEMPLATES FOR THE LOGICAL NAME TABLES AND NAMES CREATED WITHIN PROCSTRT.
283 ;
284
285         .ALIGN  QUAD
286 PROC_DIR:                               ; LNM$PROCESS_DIRECTORY TEMPLATE
287         .LONG   0                       ; FORWARD LINK
288         .LONG   0                       ; BACK LINK
289         .WORD   PROC_DIR_SIZE           ; SIZE OF STRUCTURE
290         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
291         .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
292         .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
293         .BYTE   LNMB$M_NO_ALIAS!-       ; NO ALIAS ALLOWED
294                 LNMB$M_TABLE!-          ; THIS IS A TABLE
295                 LNMB$M_NODELETE         ; ... THAT CANNOT BE DELETED
296         .ASCIC  LNM$PROCESS_DIRECTORY ; DIRECTORY NAME AS COUNTED STRING
297
298         .BYTE   LNMX$M_TERMINAL         ; TERMINAL TRANSLATION
299         .BYTE   LNMX$C_TABLE            ; SPECIAL TABLE TRANSLATION INDEX
300         .WORD   0                       ; TRANSLATION HASH CODE
301         .BYTE   LNMTH$K_LENGTH          ; SIZE OF TABLE HEADER BLOCK
```

```
  302
  303 PROC_DIR_LNMTH = . - PROC_DIR
  304         .BYTE   LNMTH$M_DIRECTORY       ; TABLE IS FOR A DIRECTORY
  305         .LONG   0                       ; ADDRESS OF HASH TABLE
  306         .LONG   0                       ; ADDRESS OF OBJECT RIGHTS BLOCK
  307         .LONG   0                       ; ADDRESS OF CONTAINING LNMB BLOCK
  308         .LONG   0                       ; ADDRESS OF PARENT TABLE
  309         .LONG   0                       ; ADDRESS OF CHILD TABLE
  310         .LONG   0                       ; ADDRESS OF SIBLING TABLE
  311         .LONG   0                       ; ADDRESS OF TABLE HOLDING QUOTA
  312         .LONG   ^X7FFFFFFF              ; INITIAL QUOTA    ( POSITIVE INFINITY )
  313         .LONG   ^X7FFFFFFF              ; REMAINING QUOTA ( POSITIVE INFINITY )
  314
  315         .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
  316         .ALIGN  QUAD
  317 PROC_DIR_SIZE = . - PROC_DIR
  318
  319 PROC_TABLE = . - PROC_DIR              ; LNM$PROCESS_TABLE TEMPLATE
  320         .LONG   0                       ; FORWARD LINK
  321         .LONG   0                       ; BACK LINK
  322         .WORD   PROC_TABLE_SIZE         ; SIZE OF STRUCTURE
  323         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
  324         .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
  325         .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
  326         .BYTE   LNMB$M_NO_ALIAS!-       ; NON-ALIASABLE
  327                 LNMB$M_TABLE            ; A TABLE
  328         .ASCIC  LNM$PROCESS_TABLE     ; TABLE NAME AS COUNTED STRING

  329
  330         .BYTE   LNMX$M_TERMINAL         ; TERMINAL TRANSLATION
  331         .BYTE   LNMX$C_TABLE            ; SPECIAL TABLE TRANSLATION INDEX
  332         .WORD   0                       ; TRANSLATION HASH CODE
  333         .BYTE   LNMTH$K_LENGTH          ; SIZE OF TABLE HEADER BLOCK
  334
  335 PROC_TABLE_LNMTH = . - PROC_DIR
  336         .BYTE   0                       ; FLAGS BYTE
  337         .LONG   0                       ; ADDRESS OF HASH TABLE
  338         .LONG   0                       ; ADDRESS OF OBJECT RIGHTS BLOCK
  339         .LONG   0                       ; ADDRESS OF CONTAINING LNMB BLOCK
  340         .LONG   0                       ; ADDRESS OF PARENT TABLE
  341         .LONG   0                       ; ADDRESS OF CHILD TABLE
  342         .LONG   0                       ; ADDRESS OF SIBLING TABLE
  343         .LONG   0                       ; ADDRESS OF TABLE HOLDING QUOTA
  344         .LONG   0                       ; INITIAL QUOTA    ( POOLED )
  345         .LONG   0                       ; REMAINING QUOTA ( POOLED )
  346
  347         .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
  348         .ALIGN  QUAD
  349 PROC_TABLE_SIZE = . - PROC_DIR - PROC_TABLE
  350
  351
  352 PROCESS = . - PROC_DIR                 ; LNM$PROCESS TEMPLATE
  353         .LONG   0                       ; FORWARD LINK
  354         .LONG   0                       ; BACK LINK
  355         .WORD   PROCESS_SIZE            ; SIZE OF STRUCTURE
  356         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
```

```
357          .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
358          .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
359          .BYTE   0                       ; FLAGS BYTE
360          .ASCIC  LNM$PROCESS             ; LOGICAL NAME AS COUNTED STRING

361
362          .BYTE   LNMX$M_TERMINAL         ; TERMINAL TRANSLATION
363          .BYTE   0                       ; TRANSLATION INDEX IS 0
364          .WORD   0                       ; TRANSLATION HASH CODE
365          .ASCIC  LNM$PROCESS_TABLE       ; TRANSLATION AS COUNTED STRING

366
367          .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
368          .ALIGN  QUAD
369 PROCESS_SIZE = . - PROC_DIR - PROCESS
370
371 GROUP = . - PROC_DIR                    ; LNM$GROUP TEMPLATE
372          .LONG   0                       ; FORWARD LINK
373          .LONG   0                       ; BACK LINK
374          .WORD   GROUP_SIZE              ; SIZE OF STRUCTURE
375          .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
376          .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
377          .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
378          .BYTE   0                       ; FLAGS BYTE
379          .ASCIC  LNM$GROUP               ; LOGICAL NAME AS COUNTED STRING

380
381          .BYTE   LNMX$M_TERMINAL         ; TERMINAL TRANSLATION
382          .BYTE   0                       ; TRANSLATION INDEX IS 0
383          .WORD   0                       ; TRANSLATION HASH CODE
384          .ASCIC  LNM$GROUP_XXXXXX        ; TRANSLATION AS COUNTED STRING

385
386          .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
387 GROUP_XEND_SIZE = . - PROC_DIR - GROUP
388          .ALIGN  QUAD
389 GROUP_SIZE = . - PROC_DIR - GROUP
390
391 JOB = . - PROC_DIR                      ; LNM$JOB TEMPLATE
392          .LONG   0                       ; FORWARD LINK
393          .LONG   0                       ; BACK LINK
394          .WORD   JOB_SIZE                ; SIZE OF STRUCTURE
395          .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
396          .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
397          .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
398          .BYTE   0                       ; FLAGS BYTE
399          .ASCIC  LNM$JOB                 ; LOGICAL NAME AS COUNTED STRING

400
401          .BYTE   LNMX$M_TERMINAL         ; TERMINAL TRANSLATION
402          .BYTE   0                       ; TRANSLATION INDEX IS 0
403          .WORD   0                       ; TRANSLATION HASH CODE
404          .ASCIC  LNM$JOB_XXXXXXXX        ; TRANSLATION AS COUNTED STRING
```

```
405
406         .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
407 JOB_XEND_SIZE = . - PROC_DIR - JOB
408         .ALIGN  QUAD
409 JOB_SIZE = . - PROC_DIR - JOB
410
411 SYS$INPUT = . - PROC_DIR                 ; SYS$INPUT TEMPLATE
412         .LONG   0                       ; FORWARD LINK
413         .LONG   0                       ; BACK LINK
414         .WORD   SYS$INPUT_SIZE          ; SIZE OF STRUCTURE
415         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
416         .BYTE   PSL$C_EXEC              ; EXECUTIVE ACCESS MODE
417         .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
418         .BYTE   0                       ; FLAGS BYTE
419         .ASCIC  SYS$INPUT              ; LOGICAL NAME AS COUNTED STRING
420
421 SYS$INPUT_LNMX = . - PROC_DIR
422         .BYTE   0                       ; TRANSLATION ATTRIBUTES
423         .BYTE   0                       ; TRANSLATION INDEX IS 0
424         .WORD   0                       ; TRANSLATION HASH CODE
425         .BYTE   LNMX$M_XEND[PQB$S_INPUT]; WORST CASE TRANSLATION AS COUNTED STRING
426
427         .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
428         .ALIGN  QUAD
429 SYS$INPUT_SIZE = . - PROC_DIR - SYS$INPUT
430
431 TT = . - PROC_DIR                       ; TT TEMPLATE
432         .LONG   0                       ; FORWARD LINK
433         .LONG   0                       ; BACK LINK
434         .WORD   TT_SIZE                 ; SIZE OF STRUCTURE
435         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
436         .BYTE   PSL$C_EXEC              ; EXECUTIVE ACCESS MODE
437         .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
438         .BYTE   0                       ; FLAGS BYTE
439         .ASCIC  TT                     ; LOGICAL NAME AS COUNTED STRING
```

```
440
441 TT_LNMX = . - PROC_DIR
442         .BYTE   0                       ; TRANSLATION ATTRIBUTES
443         .BYTE   0                       ; TRANSLATION INDEX IS 0
444         .WORD   0                       ; TRANSLATION HASH CODE
445         .BYTE   LNMX$M_XEND[PQB$S_INPUT]; WORST CASE TRANSLATION AS COUNTED STRING
446
447         .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
448         .ALIGN  QUAD
449 TT_SIZE = . - PROC_DIR - TT
450
451 SYS$OUTPUT = . - PROC_DIR               ; SYS$OUTPUT TEMPLATE
452         .LONG   0                       ; FORWARD LINK
453         .LONG   0                       ; BACK LINK
454         .WORD   SYS$OUTPUT_SIZE         ; SIZE OF STRUCTURE
455         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
456         .BYTE   PSL$C_EXEC              ; EXECUTIVE ACCESS MODE
457         .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
458         .BYTE   0                       ; FLAGS BYTE
459         .ASCIC  SYS$OUTPUT            ; LOGICAL NAME AS COUNTED STRING
460
461 SYS$OUTPUT_LNMX = . - PROC_DIR
462         .BYTE   0                       ; TRANSLATION ATTRIBUTES
463         .BYTE   0                       ; TRANSLATION INDEX IS 0
464         .WORD   0                       ; TRANSLATION HASH CODE
465         .BYTE   LNMX$M_XEND[PQB$S_OUTPUT]; WORST CASE TRANSLATION AS COUNTED STRING
```

```
466
467         .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
468         .ALIGN  QUAD
469 SYS$OUTPUT_SIZE = . - PROC_DIR - SYS$OUTPUT
470
471 SYS$ERROR = . - PROC_DIR                ; SYS$ERROR TEMPLATE
472         .LONG   0                       ; FORWARD LINK
473         .LONG   0                       ; BACK LINK
474         .WORD   SYS$ERROR_SIZE          ; SIZE OF STRUCTURE
475         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
476         .BYTE   PSL$C_EXEC              ; EXECUTIVE ACCESS MODE
477         .LONG   0                       ; CONTAINING TABLE HEADER ADDRESS
478         .BYTE   0                       ; FLAGS BYTE
479         .ASCIC  SYS$ERROR              ; LOGICAL NAME AS COUNTED STRING
480
481 SYS$ERROR_LNMX = . - PROC_DIR
482         .BYTE   0                       ; TRANSLATION ATTRIBUTES
483         .BYTE   0                       ; TRANSLATION INDEX IS 0
484         .WORD   0                       ; TRANSLATION HASH CODE
485         .BYTE   LNMX$M_XEND[PQB$S_ERROR]; WORST CASE TRANSLATION AS COUNTED STRING
486
487         .BYTE   LNMX$M_XEND             ; LAST TRANSLATION
```

```
488          .ALIGN  QUAD
489 SYS$ERROR_SIZE = . - PROC_DIR - SYS$ERROR
490
491 SYS$DISK = . - PROC_DIR                  ; SYS$DISK TEMPLATE
492          .LONG   0                      ; FORWARD LINK
493          .LONG   0                      ; BACK LINK
494          .WORD   SYS$DISK_SIZE          ; SIZE OF STRUCTURE
495          .BYTE   DYN$C_LNM              ; TYPE OF STRUCTURE
496          .BYTE   PSL$C_EXEC             ; EXECUTIVE ACCESS MODE
497          .LONG   0                      ; CONTAINING TABLE HEADER ADDRESS
498          .BYTE   0                      ; FLAGS BYTE
499          .ASCIC  SYS$DISK              ; LOGICAL NAME AS COUNTED STRING
500
501 SYS$DISK_LNMX = . - PROC_DIR
502          .BYTE   0                      ; TRANSLATION ATTRIBUTES
503          .BYTE   0                      ; TRANSLATION INDEX IS 0
504          .WORD   0                      ; TRANSLATION HASH CODE
505          .BYTE   LNMX$M_XEND[PQB$S_DISK] ; WORST CASE TRANSLATION AS COUNTED STRING
506
507          .BYTE   LNMX$M_XEND            ; LAST TRANSLATION
508          .ALIGN  QUAD
509 SYS$DISK_SIZE = . - PROC_DIR - SYS$DISK
510 P1_ALLOC_SIZE = . - PROC_DIR
511
512      .ALIGN  5
513 GROUP_TABLE:                            ; LNM$GROUP_xxxxxx TEMPLATE
514          .LONG   0                      ; FORWARD LINK
515          .LONG   0                      ; BACK LINK
516          .WORD   GROUP_TABLE_SIZE       ; SIZE OF STRUCTURE
517          .BYTE   DYN$C_LNM              ; TYPE OF STRUCTURE
518          .BYTE   PSL$C_KERNEL           ; KERNEL ACCESS MODE
519          .LONG   0                      ; CONTAINING TABLE HEADER ADDRESS
520          .BYTE   LNMB$M_NO_ALIAS!-      ; NON-ALIASABLE
521                  LNMB$M_TABLE           ; A TABLE
522          .ASCIC  LNM$GROUP_XXXXXX      ; TABLE NAME AS COUNTED STRING
```

```
523
524          .BYTE   LNMX$M_TERMINAL          ; TERMINAL TRANSLATION
525          .BYTE   LNMX$C_TABLE             ; SPECIAL TABLE TRANSLATION INDEX
526          .WORD   0                        ; TRANSLATION HASH CODE
527          .BYTE   LNMTH$K_LENGTH           ; SIZE OF TABLE HEADER BLOCK
528
529 GROUP_TABLE_LNMTH = . - GROUP_TABLE
530          .BYTE   LNMTH$M_SHAREABLE!-      ; TABLE IS SHAREABLE
531                  LNMTH$M_GROUP            ; A GROUP TABLE
532          .LONG   0                        ; ADDRESS OF HASH TABLE
533          .LONG   0                        ; ADDRESS OF OBJECT RIGHTS BLOCK
534          .LONG   0                        ; ADDRESS OF CONTAINING LNMB BLOCK
535          .LONG   0                        ; ADDRESS OF PARENT TABLE
536          .LONG   0                        ; ADDRESS OF CHILD TABLE
537          .LONG   0                        ; ADDRESS OF SIBLING TABLE
538          .LONG   0                        ; ADDRESS OF TABLE HOLDING QUOTA
539          .LONG   0                        ; INITIAL QUOTA    ( POOLED )
540          .LONG   0                        ; REMAINING QUOTA ( POOLED )
541
542          .BYTE   LNMX$M_XEND              ; LAST TRANSLATION
543
544          .ALIGN  QUAD
545 GROUP_TABLE_ORB = . - GROUP_TABLE
546          .LONG   0                        ; GROUP NUMBER + 0 MEMBER NUMBER
547          .WORD   -1, 0                    ; INITIALIZED ACL MUTEX
548          .WORD   GROUP_TABLE_ORB_SIZ      ; SIZE OF OBJECT RIGHTS BLOCK
549          .BYTE   DYN$C_ORB                ; BLOCK TYPE
550          .BYTE   0                        ; NOTE NO ACL AS YET
551          .LONG   0                        ; ZERO RESERVED WORD & REF COUNT
552          .QUAD   0                        ; OBJECT DOES NOT HAVE AN ACCESS MODE
553          .LONG   ^X00000000               ; SYSTEM PROTECTION IS RWED
554          .LONG   ^X0000001E               ; OWNER PROTECTION
555          .LONG   ^X0000001E               ; GROUP PROTECTION IS R
556          .LONG   ^X0000001F               ; WORLD PROTECTION
557          .LONG   0,0                      ; NULL INITIAL ACL
558          .BYTE   0[ORB$S_MIN_CLASS]       ; MINIMUM CLASSIFICATION MASK
559          .BYTE   0[ORB$S_MAX_CLASS]       ; MAXIMUM CLASSIFICATION MASK
560          .ALIGN  5
561 GROUP_TABLE_ORB_SIZ = . - GROUP_TABLE - GROUP_TABLE_ORB
562 GROUP_TABLE_SIZE = . - GROUP_TABLE
563
564 JOB_TABLE = . - GROUP_TABLE              ; LNM$JOB_xxxxxxxx TEMPLATE
565          .LONG   0                        ; FORWARD LINK
566          .LONG   0                        ; BACK LINK
567          .WORD   JOB_TABLE_SIZE           ; SIZE OF STRUCTURE
568          .BYTE   DYN$C_LNM                ; TYPE OF STRUCTURE
569          .BYTE   PSL$C_KERNEL             ; KERNEL ACCESS MODE
570          .LONG   0                        ; CONTAINING TABLE HEADER ADDRESS
571          .BYTE   LNMB$M_NO_ALIAS!-        ; NON-ALIASABLE
572                  LNMB$M_TABLE             ; A TABLE
573          .ASCIC  LNM$JOB_XXXXXXXX       ; TABLE NAME AS COUNTED STRING
```

```
574
575          .BYTE   LNMX$M_TERMINAL          ; TERMINAL TRANSLATION
576          .BYTE   LNMX$C_TABLE             ; SPECIAL TABLE TRANSLATION INDEX
577          .WORD   0                        ; TRANSLATION HASH CODE
578          .BYTE   LNMTH$K_LENGTH           ; SIZE OF TABLE HEADER BLOCK
579
580 JOB_TABLE_LNMTH = . - GROUP_TABLE
581          .BYTE   LNMTH$M_SHAREABLE        ; SHAREABLE TABLE
582          .LONG   0                        ; ADDRESS OF HASH TABLE
583          .LONG   0                        ; ADDRESS OF OBJECT RIGHTS BLOCK
584          .LONG   0                        ; ADDRESS OF CONTAINING LNMB BLOCK
585          .LONG   0                        ; ADDRESS OF PARENT TABLE
586          .LONG   0                        ; ADDRESS OF CHILD TABLE
587          .LONG   0                        ; ADDRESS OF SIBLING TABLE
588          .LONG   0                        ; ADDRESS OF TABLE HOLDING QUOTA
589          .LONG   0                        ; INITIAL QUOTA
590          .LONG   0                        ; REMAINING QUOTA
591
592          .BYTE   LNMX$M_XEND              ; LAST TRANSLATION
593
594          .ALIGN  QUAD
595 JOB_TABLE_ORB = . - GROUP_TABLE
596          .LONG   0                        ; OWNER OF JOB TABLE
597          .WORD   -1, 0                    ; INITIALIZED ACL MUTEX
598          .WORD   JOB_TABLE_ORB_SIZ        ; SIZE OF OBJECT RIGHTS BLOCK
599          .BYTE   DYN$C_ORB                ; BLOCK TYPE
600          .BYTE   0                        ; NOTE NO ACL AS YET
601          .LONG   0                        ; ZERO RESERVED WORD & REF COUNT
602          .QUAD   0                        ; OBJECT DOES NOT HAVE AN ACCESS MODE
603          .LONG   ^X00000000               ; SYSTEM PROTECTION IS RWED
604          .LONG   ^X00000000               ; OWNER PROTECTION IS RWED
605          .LONG   ^X0000001F               ; GROUP PROTECTION
606          .LONG   ^X0000001F               ; WORLD PROTECTION
607          .LONG   0,0                      ; NULL INITIAL ACL
608          .BYTE   0[ORB$S_MIN_CLASS]       ; MINIMUM CLASSIFICATION MASK
609          .BYTE   0[ORB$S_MAX_CLASS]       ; MAXIMUM CLASSIFICATION MASK
610          .ALIGN  5
611 JOB_TABLE_ORB_SIZ = . - GROUP_TABLE - JOB_TABLE_ORB
612 JOB_TABLE_SIZE = . - GROUP_TABLE - JOB_TABLE
613 S0_ALLOC_SIZE = . - GROUP_TABLE
614
615 ;
616 ; OFFSETS FROM R6 USED BY THE IMAGE DUMP MERGE.
617 ;
618
619          $OFFSET 0,POSITIVE,<-
620                  <,<<IMGACT$_NARGS+1>*4>>,- ; ARGUMENT LIST FOR $IMGACT CALL
621                  <IMGACT_INADR,8>,-      ; INPUT ADDRESS RANGE TO MAP IMAGE AT
622                  <IMGACT_RETADR,8>,-     ; RET ADR RANGE WHERE IMAGE WAS MAPPED
623                  <HDRBUF,512>,-          ; IMAGE HEADER BUFFER FOR $IMGACT
624                  <PROCPRIV,8>,-          ; PROCESS DEFAULT PRIVILEGES
625                  <IMAGPRIV,8>,-          ; IMAGE PRIVILEGES
626                  <PHD_FLAGS,4>,-         ; PROCESS HEADER FLAGS
627                  <JPI_PROC,12>,-         ; GETJPI LIST ITEM FOR PROCESS PRIVILEGES
628                  <JPI_IMAG,12>,-         ; GETJPI LIST ITEM FOR IMAGE PRIVILEGES
```

```
                        <JPI_FLAG,12>,-         ; GETJPI LIST ITEM FOR FLAGS
                        <JPI_END,4>,-           ; GETJPI LIST TERMINATOR
                        <SCRATCHSIZE,0>,-       ; SIZE OF AREA ADDRESS OFF OF FP
                        >
    IMGACT_INADR:
    IMGACT_RETADR:
    HDRBUF:
    PROCPRIV:
    IMAGPRIV:
    PHD_FLAGS:
    JPI_PROC:
    JPI_IMAG:
    JPI_FLAG:
    JPI_END:
    SCRATCHSIZE:
```

```
635          .SBTTL  INI$PROCSTRT_INIT - PROCSTRT INITIALIZATION ROUTINE
636 ;++
637 ; FUNCTIONAL DESCRIPTION:
638 ;       This routine acts initializes several fields in the GROUP_TABLE
639 ;       and JOB_TABLE templates that can no longer be initialized
640 ;       statically.
641 ;
642 ; ENVIRONMENT
643 ;       This routine must execute during the SWAPPER stage of initialization,
644 ;       since it initializes pagable data.
645 ;
646 ; INPUTS:
647 ;       NONE
648 ;
649 ; OUTPUTS:
650 ;       Table address fields in the GROUP_TABLE and JOB_TABLE initialized.
651 ;       R0 destroyed.
652 ;
653 ;--
654          DECLARE_PSECT   EXEC$INIT_CODE
655
656          INITIALIZATION_ROUTINE  INI$PROCSTRT_INIT
657
658 INI$PROCSTRT_INIT::
659
660 ;
661 ; First make sure we're executing in SWAPPER context, since we must be far
662 ; enough along in bootstrapping to touch pagable data.
663 ;
664          BBS     #BOOSTATE$V_SWAPPER,-   ; Are we in the SWAPPER yet?
665                  G^EXE$GL_STATE,10$
666          BICL    #INIRTN$M_NO_RECALL,(R5); We need to be called again
667          BRB     20$
668
669 ;
670 ; Now fill in various fields in GROUP_TABLE and JOB_TABLE templates.
671 ;
672 10$:     MOVL    G^LNM_AR_SYSTEM_DIR_LNMTH,R0
673                                          ; Get address of system table head.
674          MOVL    R0,W^GROUP_TABLE+LNMC$L_TBLADDR
675          MOVL    R0,W^GROUP_TABLE+GROUP_TABLE_LNMTH+LNMTH$L_PARENT
676          MOVL    R0,W^GROUP_TABLE+GROUP_TABLE_LNMTH+LNMTH$L_QTABLE
677
678          MOVL    R0,W^GROUP_TABLE+JOB_TABLE+LNMC$L_TBLADDR
679          MOVL    R0,W^GROUP_TABLE+JOB_TABLE_LNMTH+LNMTH$L_PARENT
680          MOVL    R0,W^GROUP_TABLE+JOB_TABLE_LNMTH+LNMTH$L_QTABLE
681
682 20$:     MOVL    #SS$_NORMAL,R0
683          RSB
```

```
685          .SBTTL  EXE$PROCSTRT - STARTUP NEW PROCESS
686
687 ;++
688 ; FUNCTIONAL DESCRIPTION:
689 ;
690 ; CALLING SEQUENCE:
691 ;       NONE
692 ;
693 ; INPUT PARAMETERS:
694 ;       CTL$GL_PCB - POINTS TO PCB OF CURRENT PROCESS
695 ;       PCB$L_PQB - POINTER TO PROCESS QUOTA BLOCK
696 ;
697 ; IMPLICIT INPUTS:
698 ;       IPL = IPL$_ASTDEL
699 ;
700 ; OUTPUT PARAMETERS:
701 ;       NONE
702 ;
703 ; IMPLICIT OUTPUTS:
704 ;       LOGICAL NAMES ARE DEFINED FOR 'SYS$INPUT', 'SYS$OUTPUT', AND 'SYS$ERROR'
705 ;       BASED ON THE STRINGS PASSED IN THE PROCESS QUOTA BLOCK.
706 ;
707 ; COMPLETION CODES:
708 ;       NONE
709 ;
710 ; SIDE EFFECTS:
711 ;       NONE
712 ;
713 ;--
714
715
716 ;
717 ;  The PQB address must be stored before any instruction that can cause a page
718 ;  fault. If a page fault occurs and the process is put into a resource wait
719 ;  state, then the PQB address will be lost because the EFWM field, used to
720 ;  store the resource number, overlaps PCB$L_PQB. This forces the first
721 ;  two instructions into a nonpaged program section.
722 ;
723
724          DECLARE_PSECT   EXEC$NONPAGED_CODE
725
726          UNIVERSAL_SYMBOL        EXE$PROCSTRT
727 ;EXE$PROCSTRT::                                 ; STARTUP NEW PROCESS
728          MOVL    G^CTL$GL_PCB,R4                ; GET ADDRESS OF CURRENT PCB
729          MOVL    PCB$L_PQB(R4),R6               ; GET POINTER TO PROCESS QUOTA BLOCK
730          JMP     G^EXE_PROCSTRT                 ; CONTINUE IN PAGEABLE EXEC
731
732          DECLARE_PSECT   EXEC$PAGED_CODE
733
734 EXE_PROCSTRT:
735 ;
736 ;       N O T E :   THERE CAN BE NO I/O TO A PROCESS CHANNEL BETWEEN HERE
737 ;                   AND THE END OF THE NEW CHANNEL CREATION CODE.
738 ;
739          MOVL    G^MMG$GL_RMSBASE,G^CTL$GL_RMSBASE ; SET RMS DISPATCHER BASE
740          MOVL    G^MMG$GL_CTLBASVA,G^CTL$GL_CTLBASVA ; SET CTL BASE ADDRESS
741          MOVL    G^MMG$GL_VVIEF_BASE,G^CTL$GL_VVIEF_BASE ; SET VVIEF BASE
```

```
PROCSTRT - PROCESS STARTUP AND INITIALIZATION 10-MAY-1989 16:39:04 VAX MACRO V5.0-8 Page 18
X-23 EXE$PROCSTRT - STARTUP NEW PROCESS 28-OCT-1988 08:51:22 [SYS.SRC]PROCSTRT.MAR;1 (6)

  742          MOVL    G^MMG$GL_VVIEF_END, G^CTL$GL_VVIEF_END  ; SET VVIEF END
  743          MOVL    G^MMG$GL_VVIEF_ADDR,G^CTL$GL_VVIEF_ADDR ; SET VVIEF XFER ADDR
  744 ;
  745 ; INITIALIZE THE DISPATCH VECTORS.
  746 ;
  747          MOVAB   G^CTL$A_DISPVEC,R5      ; GET ADR OF 1ST VECTOR PAGE
  748          MOVZBL  #04,NXTKVEC(R5)         ; SET OFFSET TO 1ST FREE KERNEL VECTOR
  749          MOVZBL  #04,NXTEVEC(R5)         ; SET OFFSET TO 1ST FREE EXEC VECTOR
  750          MOVZBL  #04,NXTRVEC(R5)         ; SET OFFSET TO 1ST FREE RUNDWN VECTOR
  751          MOVZBL  #04,NXTERVEC(R5)        ; SET OFFSET TO 1ST FREE EXEC MODE RUNDWN VE
  752          MOVZBL  #04,NXTMVEC(R5)         ; SET OFFSET TO 1ST FREE MESSAGE VECTOR
  753          MOVZBL  #OP$_RSB,NXTKVEC+4(R5)  ; SET AN RSB INTO THE 1ST FREE VECTOR
  754          MOVZBL  #OP$_RSB,NXTEVEC+4(R5)  ; SET AN RSB INTO THE 1ST FREE VECTOR
  755          MOVZBL  #OP$_RSB,NXTRVEC+4(R5)  ; SET AN RSB INTO THE 1ST FREE VECTOR
  756          MOVZBL  #OP$_RSB,NXTERVEC+4(R5) ; SET AN RSB INTO THE 1ST FREE VECTOR
  757          MOVAB   NXTKVEC+4(R5),G^CTL$GL_USRCHMK ; SET POINTER TO START OF VECTOR
  758          MOVAB   NXTEVEC+4(R5),G^CTL$GL_USRCHME ; SET POINTER TO START OF VECTOR
  759          MOVAB   NXTRVEC+4(R5),G^CTL$GL_USRUNDWN ; SET POINTER TO START OF VECTOR
  760          MOVAB   NXTMVEC+4(R5),G^CTL$GL_GETMSG ; SET POINTER TO START OF VECTOR
  761          MOVAB   NXTERVEC+4(R5),G^CTL$GL_USRUNDWN_EXEC ; SET POINTER TO START OF VECT
  762          ASSUME PHD$Q_PRIVMSK EQ 0
  763          MOVL    G^CTL$GL_PHD,R5         ; GET SAFE POINTER TO PROCESS HEADER WINDOW
  764          BBC     #PQB$V_IMGDMP,PQB$W_FLAGS(R6),10$
  765          BISW    #PHD$M_IMGDMP,PHD$W_FLAGS(R5) ; IMAGE DUMP WAS REQUESTED
  766
  767 ; SET UP P1 SPACE LOOKASIDE LIST FOR KERNEL MODE BUFFERS
  768
  769 10$:     MOVAB   G^CTL$GL_KRPFL,R2       ; GET LISTHEAD ADDRESS
  770          MOVAB   G^CTL$GL_KRP,R1         ; GET
  771          MOVL    S^#CTL$C_KRP_COUNT,R0
  772          BLEQ    30$
  773 20$:     INSQUE  (R1),@4(R2)
  774          ADDL    #CTL$C_KRP_SIZE,R1
  775          SOBGTR  R0,20$
  776 30$:
  777          .ENABL  LSB
  778                                          ; SET PROPER QUOTAS FROM PQB
  779          MOVL    PQB$L_CPULM(R6),PHD$L_CPULIM(R5) ; SET CPU TIME LIMIT
  780          CVTLW   PQB$L_ASTLM(R6),PHD$W_ASTLM(R5)  ; SET AST LIMIT
  781          MOVL    G^SGN$GL_MAXWSCNT,R1    ; GET MAXIMUM WORKING SET LIST LENGTH
  782          SUBL3   G^SCH$GL_FREELIM,G^PFN$GL_PHYPGCNT,R0 ; GET AVAILABLE PAGES
  783          CMPL    R0,R1                   ; MINIMIZE WITH SPECIFIED QUOTA
  784          BLEQ    100$                    ; USE QUOTA
  785          MOVL    R1,R0                   ; USE MAXIMUM WORKING SET COUNT
  786 100$:    MOVL    PQB$L_WSEXTENT(R6),R1   ; GET MAXIMUM PAGES FOR WORKING SET
  787          MOVL    PQB$L_WSQUOTA(R6),R2    ; GET MAXIMUM QUOTA FOR WORKING SET
  788          MOVL    #<1@16>,R7              ; Limit WSQUOTA to 64K pages
  789          CMPL    R2,R7                   ;*** This restriction cannot be removed
  790          BLEQU   110$                    ;*** until the swap/page file allocation
  791          MOVL    R7,R2                   ;*** routines can handle > 64K requests
  792 110$:    MOVL    PQB$L_WSDEFAULT(R6),R3  ; GET DESIRED DEFAULT
  793          CMPL    R1,R2                   ; EXTENT MUST BE BIGGER THAN QUOTA
  794          BGEQU   120$                    ; YES, USE IT AS IS
  795          MOVL    R2,R1                   ; FORCE TO QUOTA (EXTENT MAY BE 0)
  796 120$:    CMPL    R1,R0                   ; EXTENT MUST BE LESS THAN MAX PAGES
  797          BLEQU   130$                    ; BRANCH IF OK AS IS
  798          MOVL    R0,R1                   ; SET EXTENT TO MAX MEMORY
```

```
 799 130$:    CMPL    R2,R1                   ; QUOTA MUST BE LESS THAN EXTENT
 800          BLEQU   140$                    ; BRANCH IF OK AS IS
 801          MOVL    R1,R2                   ; SET QUOTA TO EXTENT
 802 140$:    CMPL    R3,R2                   ; DEFAULT MUST BE LESS THAN QUOTA
 803          BLEQU   150$                    ; BRANCH IF OK AS IS
 804          MOVL    R2,R3                   ; SET DEFAULT TO QUOTA
 805 150$:    SUBL3   #1,PHD$L_WSLIST(R5),R0  ; GET BASE OFFSET TO WORKING SET LIST
 806          ADDL    R0,R1                   ; GET EXTENT
 807          MOVL    R1,PHD$L_WSEXTENT(R5)   ; SET EXTENT
 808          MOVL    R1,PHD$L_WSAUTHEXT(R5)  ; SET AUTHORIZED EXTENT
 809          ADDL    R0,R2                   ; GET QUOTA
 810          MOVL    R2,PHD$L_WSQUOTA(R5)    ; QUOTA VALUE
 811          MOVL    R2,PHD$L_WSAUTH(R5)     ; AUTHORIZED VALUE
 812          ADDL3   R0,R3,PHD$L_DFWSCNT(R5) ; SAVE DEFAULT WORKING SET SIZE
 813
 814 ; THE AUTHPRI CELL EXISTS IN TWO PLACES. THE $SETPRI SYSTEM SERVICE USES
 815 ; THE PCB CELL BUT THE PHD CELL MUST EXIST FOREVER BECAUSE THAT IS WHERE
 816 ; THE JPI ITEM CODE BELIEVES THAT AUTHPRI IS LOCATED.
 817
 818          MOVB    PCB$B_PRIB(R4),PCB$B_AUTHPRI(R4)    ; SET INITIAL PROCESS PRIORITY
 819          MOVB    PCB$B_PRIB(R4),PHD$B_AUTHPRI(R5)    ; ... IN BOTH PCB AND PHD
 820          MOVQ    PQB$Q_PRVMSK(R6),@PCB$L_PHD(R4)     ; SET PRIVILEGES FOR PROCESS
 821          MOVQ    PQB$Q_PRVMSK(R6),G^CTL$GQ_PROCPRIV  ; BOTH PERMANENT AND CURRENT
 822          MOVQ    PQB$Q_PRVMSK(R6),PHD$Q_AUTHPRIV(R5) ; AND AUTHORIZED MASKS
 823          MOVB    PQB$B_MSGMASK(R6),G^CTL$GB_MSGMASK  ; GET DEFAULT MESSAGE FLAGS
 824          MOVQ    G^EXE$GQ_SYSTIME,G^CTL$GQ_LOGIN     ; SAVE LOGIN TIME
 825          MOVQ    R4,-(SP)                            ; SAVE PCB AND PHD POINTERS
 826
 827 ; MOVE MINIMUM AND MAXIMUM AUTHORIZED SECURITY CLEARANCE RECORDS INTO THE PHD.
 828 ; THE FOLLOWING ASSUME STATEMENTS GUARANTEE THAT WE CAN SAFELY PERFORM THIS
 829 ; WITH A SINGLE MOVC3 INSTRUCTION.
 830
 831          ASSUME PQB$S_MIN_CLASS EQ PHD$S_MIN_CLASS
 832          ASSUME PQB$S_MAX_CLASS EQ PHD$S_MAX_CLASS
 833          ASSUME PQB$R_MAX_CLASS EQ <PQB$R_MIN_CLASS + PQB$S_MIN_CLASS>
 834          ASSUME PHD$R_MAX_CLASS EQ <PHD$R_MIN_CLASS + PHD$S_MIN_CLASS>
 835
 836          MOVC3   #<PQB$S_MIN_CLASS+PQB$S_MAX_CLASS>,-
 837                  PQB$R_MIN_CLASS(R6),-
 838                  PHD$R_MIN_CLASS(R5)
 839
 840 ; INITIALIZE THE JPI CONTEXT SEGMENT QUEUE HEADER IN THE KERNEL DATA PAGE
 841
 842          MOVAB   G^CTL$GQ_JPICTX,R0      ; ADDRESS OF JPI CONTEXT QUEUE
 843          MOVL    R0,(R0)                 ; INITIALIZE FLINK
 844          MOVL    R0,4(R0)                ; ... AND BLINK
 845
 846 ; INITIALIZE THE DECLARED RESOURCE MANAGER LISTHEAD IN THE KERNEL DATA PAGE
 847
 848          MOVAB   G^CTL$GQ_RMCB_QUE,R0    ; ADDRESS OF DECLARED RM LISTHEAD
 849          MOVL    R0,(R0)                 ; INITIALIZE THE FLINK
 850          MOVL    R0,4(R0)                ; .. AND THE BLINK
 851
 852 ; INITIALIZE LISTHEADS FOR DOUBLY LINKED LISTS USED BY IMAGE ACTIVATOR
 853
 854          MOVAB   G^IAC$GL_IMAGE_LIST,R0  ; LIST OF ACTIVATED IMAGES
 855          MOVL    R0,(R0)                 ; INITIALIZE FLINK
```

```
856          MOVL    R0,4(R0)                ; ... AND BLINK
857
858          MOVAB   G^IAC$GL_WORK_LIST,R0   ; LIST OF WORK IN PROGRESS
859          MOVL    R0,(R0)                 ; INITIALIZE FLINK
860          MOVL    R0,4(R0)                ; ... AND BLINK
861
862          MOVAB   G^IAC$GL_ICBFL,R0       ; ADDRESS OF ICB LOOKASIDE LIST
863          MOVL    R0,(R0)                 ; INITIALIZE FLINK
864          MOVL    R0,4(R0)                ; ... AND BLINK
865
866 ; INITIALIZE LISTHEADS FOR CWPS QUEUES
867
868          MOVL    G^CTL$GL_PCB, R4                ; GET PCB ADDRESS
869          MOVAB   PCB$Q_CWPSSRV_QUEUE(R4), R0     ; GET CWPS QUEUE ON PCB
870          MOVL    R0,(R0)                         ; INITIALIZE FLINK
871          MOVL    R0,4(R0)                        ; ... AND BLINK
872
873          MOVL    G^CTL$GL_PHD, R4                ; GET PHD ADDRESS
874          MOVAB   PHD$Q_PSCANCTX_QUEUE(R4), R0    ; GET PSCAN QUEUE ON PHD
875          MOVL    R0,(R0)                         ; INITIALIZE FLINK
876          MOVL    R0,4(R0)                        ; ... AND BLINK
877
878 ; CREATE THE PAGES FOR THE CCB TABLE, PROCESS ALLOCATION REGION, AND DEFAULT
879 ;       IMAGE I/O SEGMENT
880
881          MOVZWL  G^SGN$GW_PCHANCNT,R3    ; PICK UP SYSGEN PARAM FOR # CHANS
882          INCL    R3                      ; ALLOW FOR WASTED CCB
883          MULL    #CCB$C_LENGTH,R3        ; CONVERT TO # BYTES
884          MOVAB   511(R3),R3              ; ROUND UP TO EVEN PAGES
885          BICL    #511,R3
886          MOVZWL  G^SGN$GW_CTLPAGES,R4    ; GET # PAGES FOR PROCESS ALL REGION
887          ASHL    #9,R4,R4                ; CONVERT TO # BYTES
888          ADDL    R4,R3                   ; GET TOTAL # BYTES NEEDED SO FAR
889          MOVZWL  G^SGN$GW_PIOPAGES,R7    ; GET # PAGES FOR PIO SEGMENT
890          ASHL    #9,R7,R7                ; CONVERT TO NUMBER OF BYTES
891          ADDL    R7,R3                   ; GET TOTAL # BYTES NEEDED
892          MOVZWL  G^SGN$GW_IMGIOCNT,R8    ; GET # PAGES FOR IIO SEGMENT
893          ASHL    #9,R8,R8                ; CONVERT TO NUMBER OF BYTES
894          ADDL    R8,R3                   ; GET TOTAL # BYTES NEEDED
895          MOVAL   G^CTL$GL_CTLBASVA,R5    ; GET POINTER TO 'TOP' OF P1
896          SUBL3   #1,(R5),-(SP)           ; 'LAST' PAGE IN P1
897          SUBL3   R3,(R5),-(SP)           ; 'TOP' OF CREATED REGION
898          MOVAQ   -(SP),R2                ; SPACE FOR RETADR
899          PUSHL   #PSL$C_KERNEL+<PRT$C_UREW@8> ; ACCESS MODE AND PROTECTION
900          PUSHL   R2                      ; RETADR ARRAY
901          PUSHAB  8(R2)                   ; INADR ARRAY
902          CALLS   #3,G^MMG$CRETVA         ; CALL INTERNAL ENTRY POINT FOR $CRETVA
903          BLBC    R0,VABUG                ; GET OUT ON ERROR
904          CMPL    (R2),8(R2)              ; DID WE GET FULL REQUEST?
905          BNEQ    VABUG                   ; NO, ERROR OUT
906          CMPL    4(R2),12(R2)            ; MAKE DOUBLY SURE
907          BEQL    DIVR                    ; NO, ERROR OUT
908
909 VABUG: MOVAB   G^EXE$GQ_PQBIQ,R1        ; ADDRESS OF PQB LIST
910          $INSQTI (R6),(R1),-(SP)         ; DEALLOCATE PQB TO LOOKASIDE LIST
911          SETIPL  #0                      ; ALLOW PROCESS TO BE DELETED
912          BRW     EXE$EXIT_IMAGE          ; DELETE THE PROCESS
```

```
913 ;
914 ; NOW DIVIDE THE CREATED SPACE INTO FOUR AREAS
915 ;
916
917 DIVR:   MOVL    (R2),G^PIO$GQ_IIODEFAULT+4                ; DEFAULT IMAGE I/O AREA
918         MOVL    R8,G^PIO$GQ_IIODEFAULT                    ; SIZE
919         MOVAL   G^PIO$GW_PIOIMPA+IMP$L_IOSEGADDR,R0       ; GET POINTER ADDRESS
920         ADDL    (R2),R8                                   ; START OF REMAINING SPACE
921         MOVL    R8,(R0)+                                  ; SET UP THE PIO SEG ADDR
922         MOVL    R7,(R0)                                   ; SET LENGTH
923         ADDL3   R8,R7,R0                                  ; GET POINTER TO FREE SPACE
924         MOVL    R0,G^CTL$GQ_ALLOCREG                      ; SET UP PROCESS ALLOCATION
925         CLRL    (R0)+                                     ; NULL FORWARD POINTER
926         MOVL    R4,(R0)                                   ; SET SIZE OF REGION
927         MOVZWL  G^SGN$GW_CTLIMGLIM,R0                     ; GET IMAGE LIMIT
928         ASHL    #9,R0,G^CTL$GL_PRCALLCNT                  ; CONVERT TO # BYTES
929         SUBL3   #CCB$C_LENGTH-1,4(R2),G^CTL$GL_CCBBASE    ; STORE BASE OF CHANNEL TABL
930
931         MOVZWL  G^SGN$GW_PCHANCNT,G^CTL$GW_NMIOCH ; SET NUMBER OF CHANNELS
932
933 ;
934 ; NOTE(!!!!):   THE ABOVE ASSIGNMENT MUST BE DONE AT THE VERY END OF THIS
935 ;               SECTION OF CODE, AS THE CELL NMIOCH BEING NON-ZERO IS AN
936 ;               INDICATOR TO G^IOC$FFCHAN THAT THERE IS ACTUALLY A REAL
937 ;               CHANNEL TABLE TO LOOK AT.
938 ;
939
940         MOVL    (R2),(R5)               ; UPDATE BASE OF VA IN CTL REGION
941         MOVAL   16(SP),SP               ; POP $CRETVA ARGS
942
943 ;
944 ; ALLOCATE P1 SPACE FOR THE PROCESS-PRIVATE LOGICAL NAME HASH TABLE, FOR
945 ; THE PROCESS DIRECTORY LOGICAL NAME TABLE, AND FOR ALL PROCESS-PRIVATE
946 ; LOGICAL NAMES AND LOGICAL NAME TABLES THAT NEED TO BE SETUP AT PROCESS
947 ; CREATION TIME. INITIALLY FORMAT THE LOGICAL NAMES AND LOGICAL NAME TABLES
948 ; BY COPYING THEIR TEMPLATES ONTO THE P1 SPACE ALLOCATED FOR THEM, AND THEN
949 ; FORMAT THE PROCESS-PRIVATE LOGICAL NAME HASH TABLE.
950 ;
951
952         MOVL    G^LNM$GL_HTBLSIZP,R1    ; RETRIEVE NUMBER OF HASH TABLE ENTRIES
953         MOVAL   @#LNMHSH$K_BUCKET[R1],R1; MULTIPLY BY 4 AND ADD OVERHEAD
954         MOVL    R1,R7                   ; SAVE SIZE OF HASH TABLE
955         MOVAB   P1_ALLOC_SIZE(R1),R1    ; ADD IN SIZE OF LOGICAL NAME BLOCKS
956         JSB     G^EXE$ALOP1PROC         ; ALLOCATE TOTAL AMOUNT OF SPACE NEEDED
957         MOVL    R2,R8                   ; SAVE ADDRESS OF ALLOCATED SPACE
958
959         MOVC5   #P1_ALLOC_SIZE,-        ; COPY TEMPLATE FOR ALL LOGICAL NAMES
960                 PROC_DIR,#0,R1,(R2)     ; AND ZERO PROCESS-PRIVATE HASH TABLE
961         MOVAB   P1_ALLOC_SIZE(R8),R3    ; COMPUTE HASH TABLE ADDRESS
962
963         MOVL    R3,G^CTL$GL_LNMHASH     ; STORE ADDRESS OF HASH TABLE AWAY
964         SUBL3   #1,G^LNM$GL_HTBLSIZP,R0 ; CALCULATE UPPER BOUND OF HASH INDEX
965         MCOML   R0,LNMHSH$L_MASK(R3)    ; STORE HASH INDEX MASK IN HASH TABLE
966         MOVW    R7,LNMHSH$W_SIZE(R3)    ; STORE HASH TABLE SIZE IN HEADER
967         MOVB    #DYN$C_RSHT,-           ; STORE HASH TABLE STRUCTURE TYPE IN
968                 LNMHSH$B_TYPE(R3)       ; HASH TABLE HEADER
969
```

```
 970 ;
 971 ; FIXUP THE PROCESS DIRECTORY LOGICAL NAME TABLE, LNM$PROCESS_DIRECTORY, AND
 972 ; LINK IT INTO THE APPROPRIATE HASH BUCKET OF THE PROCESS-PRIVATE LOGICAL NAME
 973 ; HASH TABLE.
 974 ;
 975
 976         MOVAB   PROC_DIR_LNMTH(R8),R7   ; COMPUTE DIRECTORY'S TABLE HEADER ADDR
 977         MOVL    R3,LNMTH$L_HASH(R7)     ; STORE HASH TABLE ADDR IN TABLE HEADER
 978         MOVL    R7,LNMB$L_TABLE(R8)     ; DIRECTORIES ALWAYS CONTAIN THEMSELVES
 979         MOVL    R8,LNMTH$L_NAME(R7)     ; STORE LNMB ADDRESS IN TABLE HEADER
 980         MOVL    R7,LNMTH$L_QTABLE(R7)   ; DIRECTORIES ARE QUOTA HOLDERS
 981         MOVL    R8,G^CTL$GL_LNMDIRECT   ; STORE ADDR OF PROCESS DIRECTORY AWAY
 982
 983         MOVAB   LNMB$T_NAME(R8),R1      ; RETRIEVE THE SIZE AND ADDRESS OF THE
 984         MOVZBL  (R1)+,R0                ; PROCESS DIRECTORY'S NAME
 985         JSB     G^LNM$HASH              ; HASH THE DIRECTORY NAME
 986
 987         BICL2   LNMHSH$L_MASK(R3),R0    ; MODIFY THE HASH INDEX TO BE IN RANGE
 988         MOVL    R8,LNMHSH$C_BUCKET(R3)[R0] ; INSERT THE PROCESS DIRECTORY TABLE
 989         MOVAL   LNMHSH$C_BUCKET(R3)[R0],-  ; INTO THE APPROPRIATE HASH BUCKET
 990                 LNMB$L_BLINK(R8)
 991
 992 ;
 993 ; FIXUP THE PROCESS LOGICAL NAME TABLE, LNM$PROCESS_TABLE, AND INSERT IT INTO
 994 ; THE APPROPRIATE HASH BUCKET OF THE PROCESS-PRIVATE LOGICAL NAME HASH TABLE.
 995 ;
 996
 997         MOVAB   PROC_TABLE(R8),R1       ; COMPUTE ADDRESS OF LNM$PROCESS_TABLE
 998         MOVAB   PROC_TABLE_LNMTH(R8),R9 ; COMPUTE AND SAVE ADDRESS OF LNMTH
 999         MOVL    R7,LNMB$L_TABLE(R1)     ; STORE CONTAINING TABLE HEADER'S ADDR
1000         MOVL    R3,LNMTH$L_HASH(R9)     ; STORE HASH TABLE ADDR IN TABLE HEADER
1001         MOVL    R1,LNMTH$L_NAME(R9)     ; STORE LNMB ADDRESS IN TABLE HEADER
1002         MOVL    R7,LNMTH$L_PARENT(R9)   ; LNM$PROCESS_DIRECTORY IS PARENT AND
1003         MOVL    R7,LNMTH$L_QTABLE(R9)   ; QUOTA HOLDER OF LNM$PROCESS_TABLE
1004         CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
1005         JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$PROCESS_TABLE
1006
1007 ;
1008 ; FIXUP LNM$PROCESS LNM$GROUP AND LNM$JOB AND INSERT THEM INTO THE APPROPRIATE
1009 ; HASH BUCKET OF THE PROCESS-PRIVATE LOGICAL NAME HASH TABLE. LNM$GROUP AND
1010 ; LNM$JOB REQUIRE THAT THEIR EQUIVALENCE STRINGS BE CONSTRUCTED FROM THE UIC
1011 ; AND JIB ADDRESS OF THE NEW PROCESS RESPECTIVELY.
1012 ;
1013
1014         MOVL    (SP),R4                 ; RESTORE PCB ADDRESS TO R4
1015         MOVAB   PROCESS(R8),R1          ; COMPUTE ADDRESS OF LNM$PROCESS
1016         MOVL    R7,LNMB$L_TABLE(R1)     ; STORE CONTAINING TABLE HEADER'S ADDR
1017         CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
1018         JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$PROCESS
1019
1020         MOVAB   JOB(R8),R1              ; COMPUTE ADDRESS OF LNM$JOB
1021         MOVL    R7,LNMB$L_TABLE(R1)     ; STORE CONTAINING TABLE HEADER'S ADDR
1022         MOVAB   JOB_XEND_SIZE-1(R1),R3  ; COMPUTE ADDRESS OF LAST LNMX
1023         CLRL    R2                      ; CLEAR INDEX REGISTER
1024 160$:   EXTZV   R2,#4,PCB$L_JIB(R4),R0  ; EXTRACT OUT HEX BITS AND TRANSFORM
1025         MOVB    CHARS[R0],-(R3)         ; THEM INTO THEIR ASCII EQUIVALENT
1026         ACBB    #31,#4,R2,160$          ; CONTINUE FROM RIGHT -> LEFT UNTIL DONE
```

```
1027          MOVL    R3,R10                  ; SAVE THE ADDRESS OF THE ASCII JIB ADDR
1028          CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
1029          JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$JOB
1030
1031          MOVAB   GROUP(R8),R1            ; COMPUTE ADDRESS OF LNM$GROUP
1032          MOVL    R7,LNMB$L_TABLE(R1)     ; STORE CONTAINING TABLE HEADER'S ADDR
1033          MOVAB   GROUP_XEND_SIZE-1(R1),R3; COMPUTE ADDRESS OF LAST LNMX
1034          CLRL    R2                      ; CLEAR INDEX REGISTER
1035 170$:    EXTZV   R2,#3,PCB$W_GRP(R4),R0  ; EXTRACT OUT OCTAL BITS AND TRANSFORM
1036          MOVB    CHARS[R0],-(R3)         ; THEM INTO THEIR ASCII EQUIVALENT
1037          ACBB    #14,#3,R2,170$          ; CONTINUE FROM RIGHT -> LEFT UNTIL DONE
1038          MOVB    #^A/0/,-(R3)            ; ASSUME HIGH ORDER BIT IS 0
1039          BBC     #15,PCB$W_GRP(R4),180$  ; IF SO THEN GO INSERT LNM$GROUP
1040          MOVB    #^A/1/,(R3)             ; OTHERWISE INSERT A 1
1041 180$:    MOVL    R3,R11                  ; SAVE THE ADDRESS OF THE ASCII GROUP
1042          CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
1043          JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$GROUP
1044
1045 ;
1046 ; FIXUP THE LOGICAL NAME BLOCKS FOR SYS$INPUT, TT, SYS$OUTPUT, SYS$ERROR, AND
1047 ; SYS$DISK, AND INSERT THEM INTO THE APPROPRIATE HASH BUCKET OF THE
1048 ; PROCESS-PRIVATE LOGICAL NAME HASH TABLE.
1049 ;
1050
1051          CRELNM  -                       ; FIXUP AND INSERT SYS$INPUT
1052                  PQB$T_INPUT,-
1053                  PQB$L_INPUT_ATT,-
1054                  SYS$INPUT_LNMX,-
1055                  SYS$INPUT
1056
1057          CRELNM  -                       ; FIXUP AND INSERT SYS$OUTPUT
1058                  PQB$T_OUTPUT,-
1059                  PQB$L_OUTPUT_ATT,-
1060                  SYS$OUTPUT_LNMX,-
1061                  SYS$OUTPUT
1062
1063          CRELNM  -                       ; FIXUP AND INSERT SYS$ERROR
1064                  PQB$T_ERROR,-
1065                  PQB$L_ERROR_ATT,-
1066                  SYS$ERROR_LNMX,-
1067                  SYS$ERROR
1068
1069          CRELNM  -                       ; FIXUP AND INSERT TT
1070                  PQB$T_INPUT,-
1071                  PQB$L_INPUT_ATT,-
1072                  TT_LNMX,-
1073                  TT
1074
1075          CRELNM  -                       ; FIXUP AND INSERT SYS$DISK
1076                  PQB$T_DISK,-
1077                  PQB$L_DISK_ATT,-
1078                  SYS$DISK_LNMX,-
1079                  SYS$DISK
1080
1081 ;
1082 ; IF THE PROCESS BEING CREATED IS NOT A SUB-PROCESS THEN CREATE THE JOB AND
1083 ; GROUP LOGICAL NAME TABLES.
```

```
1084 ;
1085
1086          MOVL    (SP),R4                 ; RETRIEVE PCB ADDRESS
1087          TSTL    PCB$L_OWNER(R4)         ; SUB-PROCESS?
1088          BNEQ    200$                    ; IF YES THEN SKIP TABLE CREATION
1089          MOVL    PQB$L_JTQUOTA(R6),R7    ; RETRIEVE JOB TABLE CREATION QUOTA
1090          BSBW    EXE$CRE_JGTABLE         ; CREATE JOB AND GROUP TABLES
1091          BLBS    R0,200$                 ; CONTINUE IF SUCCESS
1092 190$:    BRW     VABUG                   ; OTHERWISE, TAKE COMMON EXIT PATH
1093
1094 ;
1095 ; ALLOCATE P1 SPACE FOR THE PROCESS-PRIVATE LOGICAL NAME TABLE NAME CACHE
1096 ;
1097
1098 200$:    MULL3   #8,G^LNM$GL_HTBLSIZP,R1 ; ALLOCATE TWICE HASH TABLE SIZE
1099          DIVL3   #LNMC$K_LENGTH,R1,R8    ; COMPUTE # OF ENTRIES
1100          BEQL    220$                    ; IF ANY
1101          JSB     G^EXE$ALOP1PROC         ; ALLOCATE TOTAL AMOUNT OF SPACE NEEDED
1102          BLBC    R0,190$                 ; IF POSSIBLE
1103 210$:    MOVW    #LNMC$K_LENGTH,LNMC$W_SIZE(R2)  ; SET SIZE
1104          CLRL    LNMC$L_TBLADDR(R2)              ; MARK EMPTY
1105          INSQUE  (R2),G^CTL$GQ_LNMTBLCACHE       ; INSERT IN QUEUE
1106          MOVAB   LNMC$K_LENGTH(R2),R2            ; POINT TO NEXT
1107          SOBGTR  R8,210$                 ; LOOP
1108
1109
1110 220$:    MOVAL   G^MMG$IMGHDRBUF,AP      ; IMAGE HEADER BUFFER ADDRESS
1111
1112 ; THE FOLLOWING MOVC SEQUENCES DESTROY R0 THROUGH R5
1113
1114 IMGNAM: MOVZBL  PQB$T_IMAGE(R6),(AP)     ; SIZE OF IMAGE NAME STRING
1115          MOVAL   8(AP),4(AP)             ; ADDRESS OF IMAGE NAME STRING
1116          MOVC3   (AP),PQB$T_IMAGE+1(R6),8(AP) ; MOVE THE NAME STRING
1117
1118          TSTB    PQB$T_DDSTRING(R6)      ; CHECK FOR NULL STRING
1119          BEQL    230$                    ; YES, DONT MOVE ANYTHING
1120          MOVC3   #PQB$S_DDSTRING,-
1121                  PQB$T_DDSTRING(R6),G^PIO$GT_DDSTRING ; AND DEFAULT DIRECTORY
1122 230$:                                    ; CONTINUE
1123
1124 ; Move CLI and CLI table information to P1 space in one fell swoop:
1125 ;       PQB$T_CLI_NAME    -> G^CTL$GT_CLINAME
1126 ;       PQB$T_CLI_TABLE   -> G^CTL$GT_TABLENAME
1127 ;       PQB$T_SPAWN_CLI   -> G^CTL$GT_SPAWNCLI
1128 ;       PQB$T_SPAWN_TABLE -> G^CTL$GT_SPAWNTABLE
1129
1130          ASSUME PQB$T_CLI_TABLE   EQ <PQB$T_CLI_NAME  + PQB$S_CLI_NAME>
1131          ASSUME PQB$T_SPAWN_CLI   EQ <PQB$T_CLI_TABLE + PQB$S_CLI_TABLE>
1132          ASSUME PQB$T_SPAWN_TABLE EQ <PQB$T_SPAWN_CLI + PQB$S_SPAWN_CLI>
1133
1134          MOVC3   #<PQB$S_CLI_NAME+-
1135                    PQB$S_CLI_TABLE+-
1136                    PQB$S_SPAWN_CLI+-
1137                    PQB$S_SPAWN_TABLE>,-
1138                  PQB$T_CLI_NAME(R6),G^CTL$GT_CLINAME
1139
```

```
1140 ; STORE EVERYTHING ELSE OF INTEREST BEFORE WE GET RID OF THE PQB
1141
1142         MOVL    PQB$L_CREPRC_FLAGS(R6),G^CTL$GL_CREPRC_FLAGS
1143         MOVL    PQB$L_UAF_FLAGS(R6),G^CTL$GL_UAF_FLAGS
1144         EXTZV   #PQB$V_DEBUG,#2,-       ; Take the DEBUG flags and
1145                 PQB$W_FLAGS(R6),-       ;   place them in the process
1146                 G^CTL$GL_PSTFLAGS       ;   start flags temporarily
1147
1148 ; ****************************** TEMP ******************************
1149 ;
1150 ; THE FOLLOWING CODE WILL BE REMOVED WHEN WE DECIDE WHAT TO DO WITH THE
1151 ; ACCOUNT AND USERNAME FIELDS IN THE P1 POINTER PAGE.
1152 ;
1153         assume jib$t_account eq <jib$t_username + jib$s_username>
1154
1155         movl    G^CTL$GL_PCB,r0         ; get pcb address ...
1156         movl    pcb$l_jib(r0),r0        ; so that we can get jib address
1157         movc3   #<jib$s_username + jib$s_account>,-
1158                 jib$t_username(r0),-    ; move username and account
1159                 G^CTL$T_USERNAME        ; in one instruction
1160 ;
1161 ; **************************** END TEMP ****************************
1162
1163         MOVAB   G^EXE$GQ_PQBIQ,R1       ; ADDRESS OF PQB LIST
1164         $INSQTI (R6),(R1)               ; DEALLOCATE PQB TO LOOKASIDE LIST
1165                                         ; NOTE: R0 IS DESTROYED BY $INSQTI MACRO
1166 ;
1167 ; RESTORE PCB AND PHD ADDRESS, SET IPL TO 0 TO ALLOW FOR PROCESS DELETION
1168 ; (IF DESIRED), RESET ADDRESS SPACE, AND SET WSLAST.
1169 ;
1170         MOVL    (SP)+,R4                ; PCB ADDRESS
1171         MOVL    (SP)+,R5                ; RESTORE PHD ADDRESS
1172         JSB     G^MMG$IMGRESET          ; RESET ADDRESS SPACE AND SET WSLAST
1173
1174 ;
1175 ; INITIALIZE FIXUP VECTOR LINKED LISTS TO CONTAIN A SINGLE DUMMY ENTRY
1176 ;
1177
1178         MOVAL   G^CTL$GL_IAFPERM,G^CTL$GL_IAFLINK
1179         MOVAL   G^CTL$GL_IAFPERM,G^CTL$GL_IAFLAST
1180
1181 ;
1182 ; INITIALIZE ARRAYS THAT DETERMINE HOW PRIVILEGED VECTORS ARE RESET
1183 ;
1184         MOVAW   G^IAC$AW_VECRESET,R0    ; STORE RESET ARRAY ADDRESS
1185         MOVW    #4,(R0)+                ; KERNEL VECTOR
1186         MOVW    #4,(R0)+                ; EXEC VECTOR
1187         MOVW    #4,(R0)+                ; RUNDOWN VECTOR
1188         MOVW    #4,(R0)+                ; RUNDOWN VECTOR (Exec mode)
1189         MOVW    #4,(R0)+                ; MESSAGE VECTOR
1190
1191         MOVAW   G^IAC$AW_VECSET,R0      ; STORE START ARRAY ADDRESS
1192         MOVW    #4,(R0)+                ; KERNEL VECTOR
1193         MOVW    #4,(R0)+                ; EXEC VECTOR
1194         MOVW    #4,(R0)+                ; RUNDOWN VECTOR
1195         MOVW    #4,(R0)+                ; RUNDOWN VECTOR (Exec mode)
1196         MOVW    #4,(R0)+                ; MESSAGE VECTOR
```

```
1197
1198 ; MAP THE FILE SYSTEM INTO P1 SPACE
1199
1200          CALLS   #0, W^XQPMERGE          ; MERGE XQP INTO PROCESS
1201          BBC     S^#EXE$V_INIT, G^EXE$GL_FLAGS, -
1202                  EXE$PROCIMGACT          ; DON'T MERGE IF NOT INIT
1203          BLBC    R0, 240$                        ; EXIT IF MERGE FAILS
1204
1205          UNIVERSAL_SYMBOL        EXE$PROCIMGACT
1206 ;EXE$PROCIMGACT::                       ; ENTRY POINT FOR STAND-ALONE SYSGEN
1207          MOVL    G^CTL$GL_PCB,R4         ; GET PCB ADDRESS
1208          EXTZV   #PCB$V_HIBER,#1,PCB$L_STS(R4),R8; SAVE HIBERNATE CONTROL
1209          ROTL    #PSL$V_PRVMOD,#<PSL$C_EXEC@2+PSL$C_EXEC>,-(SP) ; FORM EXEC PSL
1210          BSBB    250$                    ; CHANGE MODE TO EXECUTIVE
1211
1212 ; ********** THE FOLLOWING CODE EXECUTES IN EXEC MODE **********
1213
1214          MOVZBL  (AP),R2                 ; GET ADR OF FILENAME STRING DESC
1215          ADDL    #3,R2                   ; ROUND THE NUMBER OF BYTES IN
1216          BICL    #3,R2                   ; THE NAME UP TO A LONGWORD BOUNDRY
1217          SUBL    R2,SP                   ; ALLOCATE SPACE FOR NAME ON STACK
1218          PUSHAB  (SP)                    ; BUILD STRING DESCRIPTOR FOR
1219          MOVZBL  (AP),-(SP)              ; FILENAME ON THE STACK
1220          MOVL    SP,R1                   ; GET ADR OF STRING DESCRIPTOR
1221          PUSHR   #^M<R1,R2,R3,R4,R5>     ; SAVE REGISTERS
1222          MOVC3   R2,@4(AP),@4(R1)        ; MOVE FILENAME TO STACK
1223          POPR    #^M<R1,R2,R3,R4,R5>     ; RESTORE REGISTERS
1224          $IMGACT_S -                     ; ACTIVATE THE IMAGE
1225                  NAME  =(AP),-           ; DESCRIPTOR FOR IMAGE NAME
1226                  DFLNAM=DEFDESC,-        ; DEFAULT NAME DESCRIPTOR
1227                  HDRBUF=(AP)             ; ADDRESS IF IMAGE HEADER BUFFER
1228          ADDL    #8,R2                   ; CALCULATE # OF BYTES ON STACK
1229          ADDL    R2,SP                   ; AND CLEAN THEM OFF
1230          BLBC    R0,240$                 ; BRANCH IF IMGACT FAILED
1231          MOVAB   G^PIO$AL_RMSEXH,R0      ; GET ADDRESS OF EXIT HANDLER CONTROL BLOCK
1232          MOVAB   W^EXE$RMSEXH,4(R0)      ; SET ADDRESS OF RMS EXIT HANDLER
1233          $DCLEXH_S (R0)                  ; DECLARE EXEC MODE EXIT HANDLER
1234 240$:    BLBC    R0,290$                 ; IF LBC ERROR
1235          MOVAB   B^EXE$CLI_UTILSRV+2,G^CTL$AL_CLICALBK ; SET CLI CALL BACK ADDRESS
1236          ROTL    #PSL$V_PRVMOD,#<PSL$C_USER@2+PSL$C_USER>,-(SP) ; FORM USER PSL
1237          BSBB    250$                    ; CHANGE TO USER MODE
1238
1239 ; ********** THE FOLLOWING CODE EXECUTES IN USER MODE **********
1240
1241          CLRL    FP                      ; TERMINATE CALL FRAME CHAIN
1242          CALLG   (AP),B^260$             ; CREATE TOP FRAME
1243 250$:    REI                             ; CHANGE TO NEW MODE
1244 260$:    .WORD   0                       ; ENTRY MASK
1245          MOVAB   B^EXE$CATCH_ALL,(FP)    ; SET EXCEPTION HANDLER ADDRESS
1246          $SETEXV_S #2,B^EXE$CATCH_ALL    ; DECLARE LAST CHANCE HANDLER
1247          $IMGFIX_S                       ; PERFORM ADDRESS RELOCATION
1248          BLBC    R0,290$                 ; QUIT IF ERROR OCCURS
1249          PUSHL   R8                      ; SAVE HIBERNATE FLAG
1250 270$:    MOVQ    (AP),R2                 ; GET IMAGE HEADER BLOCK DESCRIPTOR
1251
1252          ASSUME  CLI$V_DEBUG EQ 0
1253          ASSUME  CLI$V_DBGTRU EQ 1
```

```
1254
1255          EXTZV   #CLI$V_DEBUG,#2,-       ; Set command interpreter flags
1256                  G^CTL$GL_PSTFLAGS,-(SP) ;   from $CREPRC input.
1257
1258          PUSHL   IHD$L_LNKFLAGS(R2)      ; PUSH LINKER FLAGS
1259          MOVQ    R2,-(SP)                ; THIRD AND FOURTH ARGUMENTS TO PROG
1260          PUSHAB  B^EXE$CLI_UTILSRV       ; PUSH ADDRESS OF CLI CALL BACK ROUTINE
1261          MOVZWL  IHD$W_ACTIVOFF(R2),R1   ; OFFSET TO TRANSFER VECTOR
1262          ADDL    R1,R2                   ; FORM ADDRESS OF START VECTOR
1263          PUSHAL  (R2)                    ; MOVE TO ARGUMENT LIST
1264          BLBC    24(SP),280$             ; BR IF NO HIBERNATE
1265          $HIBER_S                        ; SET, HIBERNATE UNTIL SOME WAKE
1266 280$:    CALLS   #6,@(R2)+               ; CALL IMAGE
1267          BLBC    R0,290$                 ; EXIT IF NOT SUCCESS
1268          BLBS    (SP),270$               ; CHECK FOR HIBERNATE AGAIN
1269 290$:    BRB     EXE$EXIT_IMAGE          ;
1270
1271 ;
1272 ; DUMMY COMMAND INTERPRETER CALL BACK ROUTINE
1273 ;
1274          UNIVERSAL_ENTRY EXE$CLI_UTILSRV,-
1275                          <^M<>>
1276 ;        .ENTRY  EXE$CLI_UTILSRV,^M<>
1277          MOVL    #CLI$_INVREQTYP,R0      ; SET INVALID REQUEST TYPE STATUS
1278          RET                             ;
1279          .DSABL  LSB
```

```
1281          .SBTTL  EXIT IMAGE AND RUN DOWN FILES
1282 ;+
1283 ;
1284 ; EXE$EXIT_IMAGE - EXIT IMAGE AND RUN DOWN FILES
1285 ;
1286 ; THIS ROUTINE IS JUMPED TO AT THE CONCLUSION OF IMAGE EXECUTION TO RUN DOWN
1287 ; RMS FILES AND TO RETURN THE FINAL IMAGE STATUS.
1288 ;
1289 ; INPUTS:
1290 ;
1291 ;       R0 = FINAL IMAGE STATUS.
1292 ;
1293 ; OUTPUTS:
1294 ;
1295 ;       IMAGE EXIT IS EXECUTED.
1296 ;-
1297          UNIVERSAL_SYMBOL        EXE$EXIT_IMAGE
1298
1299 ;EXE$EXIT_IMAGE::                        ; EXIT IMAGE
1300          PUSHL   R0                      ; SAVE FINAL IMAGE STATUS
1301          PUSHL   #1                      ; SET NUMBER OF ARGUMENTS
1302 10$:     CALLG   (SP),G^SYS$EXIT         ; EXIT IMAGE
1303          BRB     10$                     ;
```

```
1305          .SBTTL  CATCH ALL CONDITION HANDLER
1306 ;+
1307 ; EXE$CATCH_ALL - CATCH ALL CONDITION HANDLER
1308 ;
1309 ; THIS ROUTINE IS ENTERED AS THE RESULT OF AN UNFIELDED OR IMPROPERLY HANDLED
1310 ; EXCEPTION CONDITION OR SOFTWARE SIGNAL.
1311 ;
1312 ; INPUTS:
1313 ;
1314 ;       CHF$L_MCHARGLST(AP) = ADDRESS OF MECHANISM ARGUMENT LIST.
1315 ;       CHF$L_SIGARGLST(AP) = ADDRESS OF CONDITION ARGUMENT LIST.
1316 ;
1317 ; OUTPUTS:
1318 ;
1319 ;       A MESSAGE IS ISSUED USING THE G^SYS$PUTMSG SYSTEM SERVICE AND A TEST IS
1320 ;       MADE ON THE CONDITION NAME TO DETERMINE IF THE IMAGE SHOULD BE ALLOWED
1321 ;       TO CONTINUE EXECUTION. THE FOLLOWING CONDITIONS CAUSE A FORCED IMAGE
1322 ;       EXIT:
1323 ;
1324 ;               1. ANY ENTRY TO THIS ROUTINE VIA THE LAST CHANCE VECTOR.
1325 ;
1326 ;               2. THE CONDITION NAME HAS A SEVERITY OF SEVERE ERROR.
1327 ;
1328 ;       IF A FORCED IMAGE EXIT IS PERFORMED, THEN A SUMMARY OF THE CONDITION
1329 ;       ARGUMENTS AND FINAL REGISTERS ARE WRITTEN TO SYS$OUTPUT.
1330 ;-
1331          UNIVERSAL_ENTRY EXE$CATCH_ALL,-
1332                          <^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>>
1333 ;        .ENTRY  EXE$CATCH_ALL,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
1334          PUSHL   #0                      ; SET EXCEPTION NAME FLAG FALSE
1335          PUSHL   R2                      ; SAVE REGISTER
1336          MOVL    CHF$L_SIGARGLST(AP),R2  ; GET ADDRESS OF SIGNAL ARGUMENTS
1337          PUSHL   (R2)                    ; SAVE NUMBER OF ARGUMENTS
1338          CMPW    CHF$L_SIG_NAME(R2),#SS$_SSFAIL  ; IS EXCEPTION SYS. SERV. FAIL.?
1339          BNEQ    100$                    ; NO
1340          $SETSFM_S       #0              ; YES, TURN OFF SYS. SERV. FAIL. EXCEP.
1341 100$:    TSTW    CHF$L_SIG_NAME+2(R2)    ; POSSIBLY SYSTEM EXCEPTION NAME?
1342          BNEQ    120$                    ; IF NEQ NO
1343          INCL    8(SP)                   ; SET EXCEPTION NAME FLAG TRUE
1344          MOVL    G^EXE$EXCEPTABLE,R1     ; GET ADDRESS OF EXCEPTION TABLE
1345          MOVZBL  (R1)+,R0                ; SET LOOP COUNT
1346 110$:    TSTB    (R1)+                   ; SKIP NUMBER OF ARGUMENTS
1347          MOVZWL  (R1)+,-(SP)             ; GET NEXT HARDWARE EXCEPTION CODE
1348          CMPZV   #STS$V_CODE,#STS$S_CODE,- ; CONDITION VALUE HARDWARE CODE?
1349                  CHF$L_SIG_NAME(R2),(SP)+ ;
1350          BEQL    130$                    ; IF EQL YES
1351          SOBGTR  R0,110$                 ; ANY MORE TO COMPARE?
1352          CLRL    8(SP)                   ; SET EXCEPTION NAME FALG FALSE
1353 120$:    SUBL    #2,(R2)                 ; ADJUST LENGTH OF ARGUMENT LIST
1354 130$:    TSTB    G^CTL$GB_SSFILTER       ; SYSTEM SERVICE INHIBITED NOW?
1355          BNEQ    140$                    ; YES, DO NOT TRY TO PRINT ANYTHING
1356          PUSHL   #0                      ; CLEAR ADDRESS OF FACILITY NAME DESCRIPTOR
1357          PUSHL   #0                      ; CLEAR ADDRESS OF ACTION ROUTINE
1358          PUSHAB  (R2)                    ; SET ADDRESS OF MESSAGE VECTOR
1359          CALLS   #3,G^SYS$PUTMSG         ; OUTPUT MESSAGE
1360 140$:    POPL    (R2)                    ; RESTORE ARGUMENT COUNT
1361          MOVL    CHF$L_SIG_NAME(R2),R0   ; GET CONDITION NAME
```

```
1362          POPL    R2                      ; RESTORE REGISTER
1363          MOVL    CHF$L_MCHARGLST(AP),R1  ; GET ADDRESS OF MECHANISM ARRAY
1364          ADDL3   #3,CHF$L_MCH_DEPTH(R1),-(SP) ; LAST CHANCE ENTRY?
1365          BEQL    160$                    ; IF EQL YES
1366          BLBS    R0,150$                 ; IF LBS SUCCESS CODE
1367          CMPZV   #STS$V_SEVERITY,#STS$S_SEVERITY,- ; SEVERE ERROR OR GREATER?
1368                  R0,#STS$K_SEVERE        ;
1369          BGEQ    160$                    ; IF GEQ YES
1370 150$:   MOVZWL  #SS$_CONTINUE,R0         ; SET CONTINUATION CODE
1371          RET                             ;
1372
1373 160$:   PUSHL   R0                       ; SAVE EXCEPTION NAME
1374          TSTB    G^CTL$GB_SSFILTER       ; SYSTEM SERVICES INHIBITED NOW?
1375          BNEQ    180$                    ; YES, DON'T TRY TO PRINT ANYTHING
1376          BLBC    8(SP),170$              ; IF LBC NOT EXCEPTION
1377          PUSHAB  (AP)                    ; SET ADDRESS OF SIGNAL ARGUMENTS
1378          PUSHAB  SUFFIX                  ; SET ADDRESS OF MESSAGE SUFFIX
1379          CALLS   #2,G^SYS$EXCMSG         ; OUTPUT EXCEPTION SUMMARY
1380 170$:   BSBW    EXE$IMGDMP_MERGE        ; TRY TO TAKE A DUMP
1381 180$:   POPL    R0                      ; RESTORE EXCEPTION NAME
1382          BBSS    #STS$V_INHIB_MSG,R0,190$        ; SET INHIBIT MESSAGE BIT
1383 190$:   BRW     EXE$EXIT_IMAGE          ;
```

```
1385          .SBTTL G^EXE$RMSEXH - EXEC Mode Exit Handler
1386 ;+
1387 ; EXE$RMSEXH - Executive mode exit handler
1388 ;
1389 ; This routine is called as the result of an attempt to exit from exec mode.
1390 ; It's function is to run down all RMS files.
1391 ;
1392 ; INPUTS:
1393 ;
1394 ;       NONE.
1395 ;
1396 ; OUTPUTS:
1397 ;
1398 ;       NONE.
1399 ;
1400 ; SIDE EFFECTS:
1401 ;
1402 ;       RMS files are run down.
1403 ;-
1404
1405          UNIVERSAL_ENTRY EXE$RMSEXH,<^M<>>
1406 ;        .ENTRY  EXE$RMSEXH,^M<>
1407          MOVAB   -128(SP),SP             ; ALLOCATE STRING BUFFER
1408          PUSHAB  (SP)                    ; BUILD BUFFER DESCRIPTOR
1409          PUSHL   #0                      ;
1410 10$:     MOVZBL  #128,(SP)               ; SET LENGTH OF STRING BUFFER
1411          PUSHL   #1                      ; RUN DOWN IMAGE AND ALL PPFS
1412          PUSHAB  4(SP)                   ; PUSH ADDRESS OF BUFFER DESCRIPTOR
1413          CALLS   #2,G^SYS$RMSRUNDWN      ; RUN DOWN THE NEXT FILE
1414          CMPL    R0,#RMS$_BUSY           ; BUSY ERROR IMPLIES DON'T TRY
1415          BEQL    20$                     ;  TO DO RUNDOWN AT ALL
1416          BLBC    R0,10$                  ; IF LBC, THEN MORE TO GO
1417 20$:     RET
1418
```

```
1420          .SBTTL XQPMERGE - Merge the XQP into P1 Space
1421 ;++
1422 ; FUNCTIONAL DESCRIPTION:
1423 ;
1424 ; This routine merges the XQP into P1 space.
1425 ;
1426 ; The number of global sections specified by G^XQP$GL_SECTIONS is mapped into
1427 ; the end of P1 space.  The sections have names of the form SYSXQP_nnn where
1428 ; nnn ranges from zero to XQP$W_SECTIONS-1.  The section is mapped writeable-
CRF
1429 ; if the corresponding bit in XQP$GL_SECPROT is set.
1430 ;
1431 ; CALLING SEQUENCE:
1432 ;
1433 ;       CALLS   #0,XQPMERGE
1434 ;
1435 ; INPUT PARAMETERS:
1436 ;
1437 ;       NONE
1438 ;
1439 ; IMPLICIT INPUT:
1440 ;
1441 ;       none
1442 ;
1443 ; OUTPUT PARAMETERS:
1444 ;
1445 ;       none
1446 ;
1447 ; IMPLICIT OUTPUT:
1448 ;
1449 ;       NONE
1450 ;
1451 ; COMPLETION CODES:
1452 ;
1453 ;       R0 low bit set => XQP successfully merged
1454 ;
1455 ;               SS$_NORMAL
1456 ;
1457 ;       R0 low bit clear => Error occurred while merging XQP
1458 ;
1459 ;               Various errors returned by $IMGACT and $MGBLSC
1460 ;
1461 ; SIDE EFFECTS:
1462 ;
1463 ;       The permanent portion of P1 space is
1464 ;       expanded to accommodate the merged image.
1465 ;
1466 ;--
1467
1468 XQPMERGE:
1469          .WORD   ^M<R2,R3,R4,R5,R6,R7>   ;REGISTER SAVE MASK
1470 ;
1471          TSTL    G^XQP$GL_DZRO           ;IS THERE ANY DZRO
1472          BEQL    10$                     ;NO
1473          $EXPREG_S -                     ;CREATE THE XQP OWN STORAGE
1474                  PAGCNT = G^XQP$GL_DZRO,-
1475                  REGION = #1,-
1476                  ACMODE = #PSL$C_EXEC
```

```
1477          BLBC    R0,30$
1478 ;
1479 10$:     SUBL    S^#<XQP_NAMSIZ+3>&^C3,SP ;RESERVE SPACE FOR GSD NAME
1480          MOVL    SP,R6                    ;SAVE ADDRESS OF GSD NAME
1481          MOVC3   S^#XQP_NAMSIZ,B^XQP_NAM,(R6) ;PUT GSD NAME IN WRITEABLE STORAGE
1482          MOVL    G^XQP$GL_SECTIONS,R3     ;COUNT OF SECTIONS TO MAP
1483          ADDB    R3,B^XQP_NAMSIZ-1(R6)    ;START WITH LAST GSD NAME
1484          PUSHL   R6                       ;BUILD DESCRIPTOR FOR GSD NAME
1485          PUSHL   S^#XQP_NAMSIZ
1486          MOVL    SP,R2                    ;ADDRESS OF DESCRIPTOR
1487          MOVL    #^X7FFFFFFF,-(SP)        ;END VA FOR BLUEPRINT P0 VA RANGE
1488          MOVL    (SP),-(SP)               ;START VA FOR BLUEPRINT P0 VA RANGE
1489          MOVL    SP,R4                    ;ADR OF INPUT VA RANGE
1490          CLRQ    -(SP)                    ;RETURN VA RANGE
1491          MOVL    SP,R5                    ;ADR OF RETURN VA RANGE
1492          DECL    R3                       ;MAKE COUNT ZERO-BASED
1493 20$:     DECB    B^XQP_NAMSIZ-1(R6)       ;NEXT GSD NAME
1494          $MGBLSC_S -
1495                  INADR  = (R4),-
1496                  RETADR = (R5),-
1497                  FLAGS  = #<SEC$M_EXPREG!SEC$M_SYSGBL>,-
1498                  GSDNAM = (R2),-
1499                  ACMODE = #PSL$C_EXEC
1500          BLBC    R0,30$
1501          SOBGEQ  R3,20$
1502 ;
1503          MOVL    (R5),G^CTL$GL_CTLBASVA   ;SET A NEW CONTROL REGION BASE
1504          JMP     @(R5)                    ;XQP SELF-INITIALIZATION
1505 30$:     RET                              ; AND RETURN TO CALLER
1506
1507 XQP_NAM:
1508          .ASCII  /SYSXQP_000/
1509          XQP_NAMSIZ = .-XQP_NAM
```

```
1511          .SBTTL  IMAGE DUMP MERGE
1512 ;+
1513 ; EXE$IMGDMP_MERGE - MERGE IN THE IMAGE DUMP FACILITY AND CALL IT
1514 ;
1515 ; THIS ROUTINE IS ENTERED AS THE RESULT OF A FATAL CONDITION WHICH WILL FORCE
1516 ; IMAGE EXIT
1517 ;
1518 ; INPUTS:
1519 ;
1520 ;       CHF$L_MCHARGLST(AP) = ADDRESS OF MECHANISM ARGUMENT LIST.
1521 ;       CHF$L_SIGARGLST(AP) = ADDRESS OF CONDITION ARGUMENT LIST.
1522 ;
1523 ; OUTPUTS:
1524 ;
1525 ;       AFTER PRIVILEGE CHECKS, THE IMAGE DUMP FACILITY IS MERGED INTO THE
1526 ;       ADDRESS SPACE AND CALLED.
1527 ;
1528 ;-
1529
1530          UNIVERSAL_SYMBOL        EXE$IMGDMP_EXEC
1531
1532 ;EXE$IMGDMP_EXEC::                          ; EXEC MODE ENTRY POINT
1533          PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
1534          BRB     EXEC_M
1535
1536          .ENABL  LSB
1537
1538          UNIVERSAL_SYMBOL        EXE$IMGDMP_MERGE
1539 ;EXE$IMGDMP_MERGE::
1540          PUSHR   #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
1541          MOVPSL  R0                          ; GET CURRENT PSL
1542          EXTZV   #PSL$V_CURMOD,#PSL$S_CURMOD,R0,R0
1543          CMPL    R0,#PSL$C_USER              ; IS IT USER MODE
1544          BNEQ    10$                         ; NO - DUMP NOT ALLOWED
1545 EXEC_M: MOVAL   -SCRATCHSIZE(SP),SP         ; RESERVE SCRATCH SPACE ON STACK
1546          MOVL    SP,R6
1547          MOVC5   #0,(SP),#0,#SCRATCHSIZE,(SP) ; ZERO IT
1548          MOVL    #<JPI$_PROCPRIV@16>+4,JPI_PROC(R6) ;INITIALIZE TO GET PROCESS PRIV
1549          MOVAB   PROCPRIV(R6),JPI_PROC+4(R6)
1550          MOVL    #<JPI$_IMAGPRIV@16>+4,JPI_IMAG(R6) ;INITIALIZE TO GET IMAGE PRIV
1551          MOVAB   IMAGPRIV(R6),JPI_IMAG+4(R6)
1552          MOVL    #<JPI$_PHDFLAGS@16>+4,JPI_FLAG(R6) ;INITIALIZE TO GET PHD FLAGS
1553          MOVAB   PHD_FLAGS(R6),JPI_FLAG+4(R6)
1554          MOVAB   JPI_PROC(R6),R0 ; ADDRESS OF ITEM LIST
1555          $GETJPI_S EFN = S^#EXE$C_SYSEFN,-
1556                    ITMLST = (R0)
1557          BLBS    R0,20$
1558 10$:     BRW 40$                             ; ERROR - GIVE UP
1559 20$:     BBC     #PHD$V_IMGDMP,PHD_FLAGS(R6),10$ ; NO DUMP REQUESTED
1560          BICL    PROCPRIV(R6),IMAGPRIV(R6) ; TEST THAT IMAGE PRIVILEGES AREN'T GREATE
1561          BICL    PROCPRIV+4(R6),IMAGPRIV+4(R6)
1562          BISL    IMAGPRIV+4(R6),IMAGPRIV(R6)
1563          BEQL    30$                         ; NO EXCESS IMAGE PRIVILEGES
1564          BBS     #PRV$V_CMKRNL,PROCPRIV(R6),30$
1565          BBC     #PRV$V_SETPRV,PROCPRIV(R6),40$  ; INSUFFICIENT PRIVILEGES
1566 30$:     MOVL    #IMGACT$_NARGS,(R6)         ; SET ARGUMENT COUNT FOR $IMGACT CALL
1567          MOVAB   IMGDMPNAM,IMGACT$_NAME(R6) ; SET ADR OF INPUT FILE NAME DESC
```

```
1568          MOVAB   DEFAULTNAMDSC,IMGACT$_DFLNAM(R6) ; SET ADR OF DEFAULT NAME STR
1569          MOVL    #<IAC$M_MERGE ! IAC$M_EXPREG>,IMGACT$_IMGCTL(R6) ; SET CTL FLAGS
1570          MOVAB   HDRBUF(R6),IMGACT$_HDRBUF(R6) ; SET ADR OF IMAGE HEADER BUFFER
1571          MOVAB   IMGACT_INADR(R6),IMGACT$_INADR(R6) ; SET ADR OF INPUT VA RANGE
1572          MOVAB   IMGACT_RETADR(R6),IMGACT$_RETADR(R6) ; SET ADR OF RETURN RANGE
1573          CLRL    IMGACT$_IDENT(R6)       ; NO MATCH IDENT SPECIFIED
1574          MOVZWL  #^X200,IMGACT_INADR(R6) ; SET A BLUEPRINT P0 ADDRESS RANGE FOR
1575          MOVL    #1@30-1,IMGACT_INADR+4(R6) ; MAPPING TO FIRST FREE VA SPACE
1576          $IMGACT_G (R6)                  ; MAP IN THE DUMP IMAGE
1577          BLBC    R0,40$                  ; ERROR - GIVE UP
1578          $IMGFIX_S
1579          BLBC    R0,40$
1580          MOVL    IMGACT_RETADR(R6),R1    ; START OF THE MERGED IN CODE
1581          ADDL3   8(R1),R1,R1             ; START ADDRESS OF THE DUMP ROUTINE
1582          MOVAL   SCRATCHSIZE(R6),SP      ; GET RID OF SCRATCH STORAGE
1583          JSB     (R1)
1584          BRB     50$
1585 ;
1586 40$:     MOVAL   SCRATCHSIZE(R6),SP      ; GET RID OF SCRATCH STORAGE
1587 50$:     POPR    #^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
1588          RSB
1589          .DSABL  LSB
1590
1591          .SBTTL CRELNM - FIXUP AND INSERT A LOGICAL NAME BLOCK
1592 ;++
1593 ; FUNCTIONAL DESCRIPTION:
1594 ;
1595 ; THE PURPOSE OF THIS ROUTINE IS TO FIXUP A LNMB FOR A LOGICAL NAME AND INSERT
1596 ; IT INTO THE APPROPRIATE HASH BUCKET OF THE PROCESS-PRIVATE LOGICAL NAME HASH
1597 ; TABLE. THE LOGICAL NAME BEING FIXED REQUIRES THAT ITS EQUIVALENCE STRING BE
1598 ; MOVED FROM THE PQB INTO THE STORAGE ALLOCATED FOR IT. IF THE LENGTH OF THE
1599 ; EQUIVALENCE STRING IS 0 THEN THE BLOCK OF STORAGE FOR THIS LOGICAL NAME IS
1600 ; DEALLOCATED AND THE ROUTINE EXITS.
1601 ;
1602 ; CALLING SEQUENCE:
1603 ;
1604 ;       BSBW    CRELNM
1605 ;       .WORD   PQB$T_<OFFSET>
1606 ;       .WORD   PQB$T_<OFFSET>_ATT
1607 ;       .WORD   <NAME>_LNMX - PROC_DIR
1608 ;       .WORD   <NAME> - PROC_DIR
1609 ;
1610 ; INPUT PARAMETERS:
1611 ;
1612 ;       R6      - ADDRESS OF PQB
1613 ;       R7      - ADDRESS OF PROCESS DIRECTORY'S TABLE HEADER
1614 ;       R8      - ADDRESS OF ALLOCATED P1 STORAGE
1615 ;       R9      - ADDRESS OF PROCESS TABLE'S TABLE HEADER
1616 ;
1617 ; IMPLICIT INPUT:
1618 ;
1619 ;       IT IS ASSUMED THAT THE LOGICAL NAME BEING CREATED HAS ALREADY BEEN
1620 ;       FORMATTED WITH THE EXCEPTION OF:
1621 ;
1622 ;       1. THE CONTAINING TABLE ADDRESS WITHIN ITS LNMB.
1623 ;       2. THE TRANSLATION STRING AND ATTRIBUTES WITHIN ITS LNMX.
1624 ;
```

```
1625 ; OUTPUT PARAMETERS:
1626 ;       NONE
1627 ;
1628 ; IMPLICIT OUTPUT:
1629 ;       NONE
1630 ;
1631 ; COMPLETION CODES:
1632 ;       NONE
1633 ;
1634 ; SIDE EFFECTS:
1635 ;
1636 ;       R0 - R3, R5, AND AP ARE DESTROYED.
1637 ;--
1638
1639 CRELNM:                                    ; FIXUP AND INSERT THE LOGICAL NAME
1640         MOVL    (SP),AP                    ; RETRIEVE ARGUEMENT POINTER
1641         ADDL2   #8,(SP)                    ; CORRECT RETURN PC VALUE
1642         MOVZWL  (AP)+,R1                   ; RETRIEVE OFFSET TO TRANSLATION
1643         MOVZBL  (R6)[R1],R0                ; RETRIEVE THE SIZE OF THE TRANSLATION
1644         BNEQ    10$                        ; IF ITS NOT 0 THEN CONTINUE
1645         MOVZWL  4(AP),R0                   ; RETRIEVE OFFSET TO LNMB
1646         ADDL2   R8,R0                      ; COMPUTE ADDRESS OF LNMB
1647         MOVZWL  LNMB$W_SIZE(R0),R1         ; RETRIEVE SIZE OF BLOCK TO DEALLOCATE
1648         JSB     G^EXE$DEAP1                ; DEALLOCATE THE LNMB
1649         BRB     20$                        ; GO RETURN
1650
1651 10$:    MOVZWL  (AP)+,R2                   ; RETRIEVE OFFSET TO TRANSLATION ATTRIBUTES
1652         MOVZWL  (AP)+,R3                   ; RETRIEVE OFFSET TO LNMX
1653         MOVB    1(R6)[R2],-                ; STORE THE TRANSLATION ATTRIBUTES FROM
1654                 LNMX$B_FLAGS(R8)[R3]       ; THE PQB INTO THEN LNMX FLAG FIELD
1655         INCL    R0                         ; MOVE COUNT ALONG WITH TRANSLATION
1656         MOVC3   R0,(R6)[R1],-              ; MOVE TRANSLATION COUNT AND STRING FROM
1657                 LNMX$T_XLATION(R8)[R3]     ; THE PQB INTO THE APPROPRIATE LNMX FIELD
1658         MOVZWL  (AP)+,R1                   ; RETRIEVE OFFSET TO LNMB
1659         ADDL2   R8,R1                      ; COMPUTE ADDRESS OF LNMB
1660         MOVL    R9,LNMB$L_TABLE(R1)        ; STORE CONTAINING TABLE HEADER'S ADDR
1661         CLRL    R2                         ; NO SPECIAL INSERTION ATTRIBUTES
1662         JSB     G^LNM$INSLOGTAB            ; APPROPRIATELY INSERT SYS$ERROR
1663 20$:    RSB                                ; RETURN
```

```
1665          .SBTTL G^EXE$CRE_JGTABLE - CREATE GROUP AND JOB-WIDE LOGICAL NAME TABLES
1666 ;++
1667 ; FUNCTIONAL DESCRIPTION:
1668 ;
1669 ; THE PURPOSE OF THIS ROUTINE IS TO HANDCRAFT GROUP AND JOB-WIDE LOGICAL
1670 ; NAME TABLES AND DIRECT THEIR INSERTION INTO THE APPROPRIATE HASH BUCKET
1671 ; OF THE SYSTEM LOGICAL NAME HASH TABLE. GROUP LOGICAL NAME TABLES ARE INSERTED
1672 ; SUCH THAT IF THERE IS AN EXISTING GROUP TABLE FOR THAT GROUP, THE CALLER OF
1673 ; THIS ROUTINE IS MAPPED TO IT.
1674 ;
1675 ; CALLING SEQUENCE:
1676 ;
1677 ;       JSB     G^EXE$CRE_JGTABLE
1678 ;
1679 ; INPUT PARAMETERS:
1680 ;
1681 ;       R7      - JOB TABLE QUOTA
1682 ;       R10     - ADDRESS OF ASCII EQUIVALENT OF JIB ADDRESS
1683 ;       R11     - ADDRESS OF ASCII EQUIVALENT OF GROUP NUMBER
1684 ;
1685 ; IMPLICIT INPUT:
1686 ;
1687 ;       LNM_AR_SYSTEM_DIR_LNMTH - ADDRESS OF POINTER TO SYSTEM DIRECTORY TABLE
1688 ;                                 HEADER
1689 ;
1690 ;       G^LNM$AL_HASHTBL        - ADDRESS OF POINTER TO SYSTEM HASH TABLE
1691 ;
1692 ; OUTPUT PARAMETERS:
1693 ;       NONE
1694 ;
1695 ; IMPLICIT OUTPUT:
1696 ;
1697 ;       R4                      - ADDRESS OF PCB
1698 ;
1699 ; COMPLETION CODES:
1700 ;
1701 ;       1               - SUCCESS
1702 ;       SS$_EXLNMQUOTA  - INSUFFICIENT QUOTA IN SYSTEM DIRECTORY TABLE
1703 ;       SS$_INSFMEM     - INSUFFICIENT PAGED POOL TO ALLOCATE LNMBS
1704 ;
1705 ; SIDE EFFECTS:
1706 ;
1707 ;       R0 - R5 AND R8 ARE DESTROYED.
1708 ;
1709 ;       THE JOB-WIDE LOGICAL NAME TABLE WILL HAVE BEEN CREATED POTENTIALLY
1710 ;       RESULTING IN THE DELETION OF ANY SHAREABLE TABLE WITH THE SAME NAME.
1711 ;
1712 ;       THE GROUP LOGICAL NAME TABLE WILL HAVE BEEN CREATED PROVIDED A GROUP
1713 ;       TABLE WITH THAT NAME DOES NOT ALRADY EXIST IN WHICH CASE NOTHING IS
1714 ;       DONE.
1715 ;
1716 ;--
1717
1718          .ENABL  LSB
1719
1720          UNIVERSAL_SYMBOL        EXE$CRE_GTABLE
1721 ;EXE$CRE_GTABLE::
```

```
1722
1723 ;
1724 ; THIS ROUTINE G^EXE$CRE_GTABLE IS IDENTICAL TO THE ROUTINE G^EXE$CRE_JGTABLE
1725 ; WITH EXCEPTION THAT THE JOB LOGICAL NAME TABLE IS NOT CREATED. THUS THE
1726 ; ONLY INPUT PARAMETER IS R11, WHICH HAS THE ADDRESS OF ASCII EQUIVALENT
1727 ; OF GROUP NUMBER.
1728 ;
1729
1730         MOVZWL  #GROUP_TABLE_SIZE,R1    ; SET SIZE OF GROUP TABLE TO BE CREATED
1731         JSB     G^EXE$ALOPAGED          ; ALLOCATE REQUIRED AMOUNT OF PAGED POOL
1732         BLBS    R0,100$                 ; CONTINUE IF ALLOCATION IS SUCCESSFUL
1733         MOVZWL  #SS$_INSFMEM,R0         ; OTHERWISE SETUP R0 WITH ERROR CODE
1734         BRW     170$                    ; AND EXIT
1735
1736 100$:   MOVL    G^CTL$GL_PCB,R4         ; GET CURRENT PCB
1737         JSB     G^LNM$LOCKW             ; LOCK LOGICAL NAME MUTEX FOR WRITING
1738
1739         MOVL    G^LNM_AR_SYSTEM_DIR_LNMTH,R8
1740                                         ; GET ADDRESS OF SYSTEM DIRECTORY TABLE
1741         CMPL    R1,LNMTH$L_BYTES(R8)    ; IS THERE ENOUGH QUOTA IN THE SYSTEM
1742                                         ; DIRECTORY TABLE?
1743         BLEQ    110$                    ; CONTINUE IF ENOUGH QUOTA
1744         MOVL    R2,R0                   ; SETUP TO DEALLOCATE THE PAGED POOL
1745         JSB     G^EXE$DEAPGDSIZ         ; DEALLOCATE IT
1746         JSB     G^LNM$UNLOCK            ; UNLOCK THE LOGICAL NAME MUTEX
1747         MOVZWL  #SS$_EXLNMQUOTA,R0      ; SETUP REASON FOR PREMATURE TERMINATION
1748         BRW     170$                    ; AND GO RETURN TO CALLER
1749
1750 110$:   MOVL    R2,R8                   ; SAVE ADDRESS OF STORAGE ALLOCATED
1751         MOVC3   R1,GROUP_TABLE,(R2)     ; FORMAT THE LOGICAL NAME TABLE(S)
1752         MOVL    G^CTL$GL_PCB,R4         ; GET CURRENT PCB
1753         BRW     CREATE_GTABLE           ; GO CREATE GROUP TABLE
1754
1755         UNIVERSAL_SYMBOL        EXE$CRE_JGTABLE
1756 ;EXE$CRE_JGTABLE::
1757
1758 ;
1759 ; ALLOCATE PAGED POOL FOR THE GROUP AND JOB-WIDE LOGICAL NAME TABLES. AFTER
1760 ; ALLOCATING SUFFICIENT PAGED POOL, WRITE LOCK THE LOGICAL NAME MUTEX, AND MAKE
1761 ; SURE THAT THE PARENT LOGICAL NAME TABLE, THE SYSTEM DIRECTORY TABLE, HAS
1762 ; SUFFICIENT QUOTA FOR THE CREATION OF BOTH LOGICAL NAME TABLES AND FOR ANY
1763 ; SEPARATE QUOTA THAT WILL BE RELEGATED TO THEM. IF THE SYSTEM DIRECTORY TABLE
1764 ; DOES NOT CONTAIN SUFFICIENT QUOTA THEN EXIT IMMEDIATELY WITH THE APPROPRIATE
1765 ; ERROR; OTHERWISE, THE BLOCK OF STORAGE THAT HAS BEEN ALLOCATED FOR THE LOGICAL
1766 ; NAME TABLES IS FORMATED.
1767 ;
1768
1769         MOVZWL  #SO_ALLOC_SIZE,R1       ; ASSUME BOTH TABLES WILL BE CREATED
1770         JSB     G^EXE$ALOPAGED          ; ALLOCATE REQUIRED AMOUNT OF PAGED POOL
1771         BLBS    R0,120$                 ; CONTINUE IF ALLOCATION IS SUCCESSFUL
1772         MOVZWL  #SS$_INSFMEM,R0         ; OTHERWISE SETUP R0 WITH ERROR CODE
1773         BRB     130$                    ; AND EXIT
1774
1775 120$:   MOVL    G^CTL$GL_PCB,R4         ; GET CURRENT PCB
1776         JSB     G^LNM$LOCKW             ; LOCK LOGICAL NAME MUTEX FOR WRITING
1777
1778         ADDL3   R7,R1,R0                ; DETERMINE TOTAL AMOUNT OF QUOTA
```

```
1779          MOVL    G^LNM_AR_SYSTEM_DIR_LNMTH,R8
1780                                          ; GET ADDRESS OF SYSTEM DIRECTORY TABLE
1781          CMPL    R0,LNMTH$L_BYTES(R8)    ; IS THERE ENOUGH QUOTA IN THE SYSTEM
1782                                          ; DIRECTORY TABLE?
1783          BLEQ    140$                    ; CONTINUE IF ENOUGH QUOTA
1784          MOVL    R2,R0                   ; SETUP TO DEALLOCATE THE PAGED POOL
1785          JSB     G^EXE$DEAPGDSIZ         ; DEALLOCATE IT
1786          JSB     G^LNM$UNLOCK            ; UNLOCK THE LOGICAL NAME MUTEX
1787          MOVZWL  #SS$_EXLNMQUOTA,R0      ; SETUP REASON FOR PREMATURE TERMINATION
1788 130$:    BRW     170$                    ; AND GO RETURN TO CALLER
1789
1790 140$:    MOVL    R2,R8                   ; SAVE ADDRESS OF STORAGE ALLOCATED
1791          MOVC3   R1,GROUP_TABLE,(R2)     ; FORMAT THE LOGICAL NAME TABLE(S)
1792          MOVL    G^CTL$GL_PCB,R4         ; GET CURRENT PCB
1793
1794 ;
1795 ; FIXUP THE LOGICAL NAME BLOCK FOR THE JOB TABLE THAT IS BEING CREATED, AND
1796 ; THEN INSERT IT INTO THE APPROPRIATE HASH BUCKET OF THE SHAREABLE LOGICAL NAME
1797 ; HASH TABLE. THIS FIXING UP OF THE JOB TABLE'S LOGICAL NAME BLOCK INCLUDES
1798 ; APPENDING TO THE LNM$JOB_ ASCII STRING ALREADY PRESENT WITHIN THE NAME FIELD
1799 ; OF THE LNMB, THE ASCII EQUIVALENT OF THE JIB'S HEXADECIMAL ADDRESS. A POINTER
1800 ; TO THIS ASCII EQUIVALENCE IS PASSED TO THIS ROUTINE IN R7.
1801 ;
1802
1803          MOVAB   JOB_TABLE(R8),R1        ; RETRIEVE ADDRESS OF JOB TABLE'S LNMB
1804          MOVQ    (R10),LNMB$T_NAME+9(R1) ; MOVE ASCII HEX JIB ADDR INTO NAME FIELD
1805          MOVL    G^LNM$AL_HASHTBL,R2     ; POINTER TO HASH TABLE
1806          MOVL    (R2),-                  ; MOVE THE ADDRESS OF THE SHAREABLE
1807                  JOB_TABLE_LNMTH+-       ; LOGICAL NAME HASH TABLE INTO THE JOB
1808                  LNMTH$L_HASH(R8)        ; TABLE'S TABLE HEADER
1809          MOVAB   JOB_TABLE_ORB(R8),-     ; MOVE THE ADDRESS OF THE JOB TABLE'S
1810                  JOB_TABLE_LNMTH+-       ; OBJECT RIGHTS BLOCK INTO THE JOB
1811                  LNMTH$L_ORB(R8)         ; TABLE'S TABLE HEADER
1812          MOVL    R1,JOB_TABLE_LNMTH+-    ; MOVE THE ADDRESS OF THE JOB TABLE'S
1813                  LNMTH$L_NAME(R8)        ; LNMB INTO THE JOB TABLE'S TABLE HEADER
1814
1815          MOVL    PCB$L_UIC(R4),-         ; MOVE THE PROCESS'S UIC INTO THE
1816                  JOB_TABLE_ORB+-         ; APPROPRIATE FIELD OF THE JOB TABLE'S
1817                  ORB$L_OWNER(R8)         ; OBJECT RIGHTS BLOCK
1818          CLRQ    JOB_TABLE_ORB+-         ; SET INITIAL NULL ACL
1819                  ORB$L_ACL_COUNT(R8)
1820
1821          TSTL    R7                      ; IS JOB TABLE QUOTA POOLED?
1822          BEQL    150$                    ; IF SO THEN GO INSERT LNMB
1823          MOVAB   JOB_TABLE_LNMTH(R8),-   ; OTHERWISE SET UP THE JOB TABLE'S
1824                  JOB_TABLE_LNMTH+-       ; TABLE HEADER AS THE QUOTA HOLDER FOR
1825                  LNMTH$L_QTABLE(R8)      ; THE JOB TABLE
1826          MOVL    R7,JOB_TABLE_LNMTH+-    ; SET THE BYTE LIMIT FIELD TO THE
1827                  LNMTH$L_BYTESLM(R8)     ; INITIAL AMOUNT OF JOB TABLE QUOTA
1828          MOVL    R7,JOB_TABLE_LNMTH+-    ; SET THE BYTE REMAINING FIELD TO THE
1829                  LNMTH$L_BYTES(R8)       ; INITIAL AMOUNT OF JOB TABLE QUOTA
1830 150$:    CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
1831          JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$GROUP_xxxxxx
1832
1833 ;
1834 ; FIXUP THE LOGICAL NAME BLOCK FOR THE GROUP TABLE THAT IS BEING CREATED, AND
1835 ; THEN INSERT IT INTO THE APPROPRIATE HASH BUCKET OF THE SHAREABLE LOGICAL NAME
```

```
1836 ; HASH TABLE PROVIDED A TABLE FOR THAT GROUP DOES NOT ALREADY EXIST. THIS
1837 ; FIXING UP OF THE GROUP TABLE'S LOGICAL NAME BLOCK INCLUDES APPENDING TO THE
1838 ; LNM$GROUP_ ASCII STRING ALREADY PRESENT WITHIN THE NAME FIELD OF THE LNMB,
1839 ; THE ASCII EQUIVALENT OF THE OCTAL GROUP THE PROCESS BELONGS TO. A POINTER
1840 ; TO THIS ASCII EQUIVALENCE IS PASSED TO THIS ROUTINE IN R8.
1841 ;
1842
1843 CREATE_GTABLE:                          ;
1844         MOVL    R8,R1                   ; SETUP TO INSERT THE GROUP TABLE'S LNMB
1845         MOVL    (R11),LNMB$T_NAME+11(R1); APPEND THE ASCII OCTAL GROUP TO THE
1846         MOVW    4(R11),LNMB$T_NAME+15(R1); LNM$GROUP_ ALREADY IN THE NAME FIELD
1847         MOVL    G^LNM$AL_HASHTBL,R2     ; ADDRESS OF HASH TABLE
1848         MOVL    (R2),-                  ; MOVE THE ADDRESS OF THE SHAREABLE
1849                 GROUP_TABLE_LNMTH+-     ; LOGICAL NAME HASH TABLE INTO THE GROUP
1850                 LNMTH$L_HASH(R8)        ; LOGICAL NAME TABLE'S YABLE HEADER
1851         MOVAB   GROUP_TABLE_ORB(R8),-   ; MOVE THE ADDRESS OF THE GROUP TABLE'S
1852                 GROUP_TABLE_LNMTH+-     ; OBJECT RIGHTS BLOCK INTO THE GROUP
1853                 LNMTH$L_ORB(R8)         ; TABLE'S TABLE HEADER
1854         MOVL    R1,GROUP_TABLE_LNMTH+-  ; MOVE THE ADDRESS OF THE LNMB INTO THE
1855                 LNMTH$L_NAME(R8)        ; GROUP TABLE'S TABLE HEADER
1856
1857         MOVW    PCB$W_GRP(R4),-         ; MOVE THE PROCESS'S GROUP NUMBER
1858                 GROUP_TABLE_ORB+-       ; INTO THE APPROPRIATE FIELD OF THE
1859                 ORB$L_OWNER+2(R8)       ; GROUP TABLE'S OBJECT RIGHTS BLOCK
1860         CLRQ    GROUP_TABLE_ORB+-       ; SET INITIAL NULL ACL
1861                 ORB$L_ACL_COUNT(R8)
1862
1863         MOVL    #LNM$M_CREATE_IF,R2     ; GROUP TABLES ARE INSERTED CREATE_IF
1864         JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$GROUP_xxxxxx
1865
1866         CMPW    #SS$_NORMAL,R0          ; DID THE GROUP TABLE ALREADY EXIST?
1867         BNEQ    160$                    ; GO UNLOCK THE MUTEX IF IT DIDN'T
1868         MOVL    R8,R0                   ; DELETE THE LNMB FOR WHAT WOULD HAVE
1869         JSB     G^EXE$DEAPAGED          ; BECOME A GROUP LOGICAL NAME TABLE
1870 ;
1871 ; UNLOCK THE LOGICAL NAME MUTEX AND RETURN STATUS.
1872 ;
1873
1874 160$:   JSB     G^LNM$UNLOCK            ; UNLOCK THE LOGICAL NAME MUTEX
1875         MOVZBL  #1,R0                   ; SUCCESS
1876 170$:   RSB                             ; RETURN
1877
1878         .DSABL  LSB
1879
1880         .END
```

15 SYSDELPRC.LIS

```
SYSDELPRC - DELETE PROCESS SYSTEM SERVICE 10-MAY-1989 16:40:14 VAX MACRO V5.0-8 Page 0
Table of contents
```

```
SYSDELPRC - DELETE PROCESS SYSTEM SERVICE 10-MAY-1989 16:40:14 VAX MACRO V5.0-8 Page 1
X-33 20-DEC-1988 12:59:50 [SYS.SRC]SYSDELPRC.MAR;1 (1)

     1          .TITLE  SYSDELPRC - DELETE PROCESS SYSTEM SERVICE
     2          .IDENT  'X-33'
     3
     4 ;
     5 ;*****************************************************************************
     6 ;*                                                                           *
     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984, 1988 BY                            *
     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
     9 ;*  ALL RIGHTS RESERVED.                                                     *
    10 ;*                                                                           *
    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
    16 ;*  TRANSFERRED.                                                             *
    17 ;*                                                                           *
    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
    20 ;*  CORPORATION.                                                             *
    21 ;*                                                                           *
    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
    24 ;*                                                                           *
    25 ;*                                                                           *
    26 ;*****************************************************************************
    27
    28 ;++
    29 ; FACILITY: EXECUTIVE, SYSTEM SERVICES
    30 ;
    31 ; ABSTRACT:     SYSDELPRC IMPLEMENTS THE DELETE PROCESS SYSTEM SERVICE
    32 ;       WHICH CAUSES A PROCESS TO BE DELETED FROM THE SYSTEM AFTER
    33 ;       RELEASING ALL OF ITS RESOURCES.
    34 ;
    35 ; ENVIRONMENT:
    36 ;       MODE=KERNEL
    37 ;
    38 ; AUTHOR:       R. I. HUSTVEDT, CREATION DATE: 30-DEC-76
    39 ;
    40 ; MODIFIED BY:
    41 ;
    42 ;       X-33    CWH5133         CW Hobbs                11-Dec-1988
    43 ;               Rearrange $DELPRC code so that we do not call
    44 ;               EXE$NAMPID at IPL$_ASTDEL.  This means a minor
    45 ;               shuffle to allocate the ACB on the NAMPID
    46 ;               success path only.
    47 ;
    48 ;       X-32    CWH5132         CW Hobbs                11-Nov-1988
    49 ;               Deallocate ACB if remote process.
    50 ;
    51 ;       X-31    CWH5131         CW Hobbs                28-Oct-1988
    52 ;               Use final names for CWPS queues.
    53 ;
    54 ;       X-30    WMC0030         Wayne Cardoza           02-Sep-1988
    55 ;               Add CPBDEF
    56 ;
    57 ;       X-29    WMC0028         Wayne Cardoza           22-Aug-1988
```

```
                ;               New affinity checking.
                ;
                ;       X-28    DOS2001         Dan O'Shaughnessy       8-MAY-1988
                ;               CWH5128         CW Hobbs                29-Aug-1988
                ;               Add support for cluster-wide process services.
                ;
                ;       X-27    SSA0006         Stan Amway              19-May-1988
                ;               Check for INPROG flag while scanning PSTs. If a PST
                ;               has this flag set, then the section was being created
                ;               at the time the delete process AST was delivered.
                ;               In this case, the REFCNT is biased by 1, which if
                ;               not handled correctly will cause a FILCNTNONZ bugcheck.
                ;
                ;       X-26    JWT0306         Jim Teague              12-Mar-1988
                ;               Now that DIOCNT bug has been found, make the bugcheck
                ;               added by X-25 a nonfatal bugcheck.
                ;
                ;       X-25    JWT0303         Jim Teague              18-Dec-1987
                ;               Permanent check for DIOCNT > DIOLM.
                ;
                ;       X-24    SJF             Stu Farnham             10-Nov-1987
                ;               Clear all affinities on process deletion.
                ;
                ;       X-23    WMC0023         Wayne Cardoza           5-Nov-1987
                ;               Reverse order of P1 deletion.
                ;
                ;       X-22    MSH0326         Michael S. Harvey       21-Oct-1987
                ;               Use interrupt stack only version of FIND_CPU_DATA
                ;               where it's appropriate.
                ;
                ;       X-21    SSA0005         Stan Amway               9-Sep-1987
                ;               Call EXE$RESETVEC1 instead of EXE$RESETVEC.
                ;               (Done on behalf of and at the request of Jon Callas.)
                ;
                ;       X-20    JWT0297         Jim Teague              20-Aug-1987
                ;               Use ADAWI for manipulating JIB$W_PRCCNT.
                ;
                ;       X-19    SSA0004         Stan Amway              10-Aug-1987
                ;               Swap file allocation changes - don't deallocate space here.
                ;
                ;       X-18    WMC0018         Wayne Cardoza           30-Jul-1987
                ;               Honor NO_DELETE bit in PCB.
                ;
                ;       X-17    SF00017         Stephen Fiorelli        27-Jul-1987
                ;               Large working set support.
                ;
                ;       X-16    WMC0016         Wayne Cardoza           22-May-1987
                ;               Fix assumption that all permanently locked in working set
                ;               pages are at high end of address space.
                ;
                ;       X-15    SSA0003         Stan Amway              13-Apr-1987
                ;               Call a new routine, MMG$DEALLOCSWPAREA, to return
                ;               swap file space.
                ;
                ;       X-14    WMC0014         Wayne Cardoza           30-Mar-1987
                ;               Make sure we zero skipped PTEs in DEL_PAGES.
                ;
```

```
115 ;       X-13    WMC0013         Wayne Cardoza           30-Mar-1987
116 ;               In DEL_PAGES, must make sure the PTE pointer is incremented
117 ;               after scanning null entries.
118 ;
119 ;       X-12    SF00003         Stephen Fiorelli        13-Mar-1987
120 ;               Happy Friday the 13th.  Store address of nullpcb
121 ;               in unused vector.
122 ;
123 ;       X-11    WMC0011         Wayne Cardoza           05-Mar-1987
124 ;               Fix a broken branch.
125 ;
126 ;       X-10    WMC0010         Wayne Cardoza           11-Feb-1987
127 ;               Release page table pages at process deletion instead of
128 ;               in swapper.
129 ;               Change NULLPCB to pointer.
130 ;
131 ;       X-7     SSA0002         Stan Amway              23-Oct-1986
132 ;               Protect exec mode rundown against re-entrancy and
133 ;               exceptions, bug checks, etc.
134 ;
135 ;       X-6     RNG0006         Rod Gamache             21-Oct-1986
136 ;               Use SMP Poor Man's Lockdown macros.
137 ;
138 ;       X-5     SSA0001         Stan Amway               2-Oct-1986
139 ;               Add support for exec mode rundown handlers.
140 ;
141 ;       X-4     HH0218          Hai Huang               08-Sep-1986
142 ;               Transfer device ownership from subprocess to job master
143 ;               for private voluems.
144 ;
145 ;       X-1D2   SF04002         Stephen Fiorelli        12-Mar-1986
146 ;               Take G^ out of LNM_SYSTEM_DIRECTORY_DESC.
147 ;
148 ;       V04-001 SF04001         Stephen Fiorelli        23-Oct-1985
149 ;               System_service macro used to declare entry point
150 ;               and build system service descriptor block.
151 ;               Added $SYSVECTORDEF.
152 ;
153 ;       V04-001 TCM0001         Trudy C. Matthews       23-Aug-1985
154 ;               Put .ASCID directives into a read/write psect.
155 ;
156 ;--
```

```
158          .SBTTL  DECLARATIONS
159 ;
160 ; INCLUDE FILES:
161 ;
162
163          $ACBDEF                         ; DEFINE AST CONTROL BLOCK OFFSETS
164          $ACCDEF                         ; DEFINE TERMINATION MESSAGE OFFSETS
165          $ACMDEF                         ; DEFINE ACCOUNTING MESSAGE OFFSETS
166          $ARBDEF                         ; DEFINE ACCESS RIGHTS BLOCK
167          $CCBDEF                         ; DEFINE CHANNEL CONTROL BLOCK
168          $CPBDEF                         ; CAPABILITIES
169          $CPUDEF                         ; DEFINE PER-CPU DATA BLOCK OFFSETS
170          $CWPSDEF                        ; DEFINE CLUSTER-WIDE SERVICE DEFINITIONS
171          $DEVDEF                         ; DEFINE DEVICE CHARACTERISTIC DEFINITIONS
172          $DYNDEF                         ; Dynamic structure definitions
173          $IODEF                          ; DEFINE I/O FUNCTION CODES
174          $IPLDEF                         ; DEFINE INTERRUPT PRIORITY LEVELS
175          $JIBDEF                         ; DEFINE JOB INFORMATION BLOCK OFFSETS
176          $MSGDEF                         ; DEFINE SYSTEM MESSAGE TYPE CODES
177          $MTLDEF                         ; DEFINE MOUNT LIST ENTRY OFFSETS
178          $OPDEF                          ; Define opcode symbolic constants
179          $PCBDEF                         ; DEFINE PCB OFFSETS
180          $PFNDEF                         ; DEFINE PFN CONSTANTS
181          $PHDDEF                         ; DEFINE PROCESS HEADER OFFSETS
182          $PRDEF                          ; DEFINE PROCESSOR REGISTER NAMES
183          $PRIDEF                         ; DEFINE PRIORITY INCREMENTS
184          $PTEDEF                         ; DEFINE PTE FIELDS
185          $PSLDEF                         ; DEFINE PSL FIELDS AND VALUES
186          $RSNDEF                         ; DEFINE RESOURCE NUMBERS
187          $SECDEF                         ; DEFINE SECTION TABLE ENTRY
188          $SSDEF                          ; DEFINE SYSTEM SERVICE STATUS CODES
189          $SYSVECTORDEF                   ; DEFINE SYSTEM SERVICE VECTOR OFFSETS
190          $UCBDEF                         ; DEFINE UNIT CONTROL BLOCK OFFSETS
191          $VCBDEF                         ; DEFINE VOLUME CONTROL BLOCK OFFSETS
192
193 ;
194 ; MACROS:
195 ;
196
197 ;
198 ; EQUATED SYMBOLS:
199 ;
200
201 ACB_L_CPULIM = ACB$L_KAST + 4
202 ACB_L_CPUTIM = ACB_L_CPULIM + 4
203 JOB_STRING_LEN = 16
204
205 ;
206 ; OWN STORAGE:
207 ;
208
209          DECLARE_PSECT   EXEC$PAGED_DATA
210
211 LNM_SYSTEM_DIRECTORY_DESC:              ; Descriptor of System Directory Table
212          .ASCID  /LNM$SYSTEM_DIRECTORY/
```

```
SYSDELPRC - DELETE PROCESS SYSTEM SERVICE 10-MAY-1989 16:40:14 VAX MACRO V5.0-8 Page 5
X-33 DECLARATIONS 20-DEC-1988 12:59:50 [SYS.SRC]SYSDELPRC.MAR;1 (2)

  213
  214 JOB_FAO:                                    ; $FAO control string descriptor for
  215         .ASCID  /LNM$JOB_!XL/               ; construction of job-wide table name
  216
  217         DECLARE_PSECT   EXEC$NONPAGED_CODE
```

```
219          .SBTTL  EXE$DELPRC - DELETE PROCESS SYSTEM SERVICE
220 ;++
221 ; FUNCTIONAL DESCRIPTION:
222 ;       EXE$DELPRC IMPLEMENTS THE DELETE PROCESS SYSTEM SERVICE WHICH CAUSES
223 ;       A PROCESS TO BE DELETED FROM THE SYSTEM.  IF THE SPECIFIED
224 ;       PROCESS EXISTS, A KERNEL AST IS QUEUED FOR IT TO
225 ;       PERFORM THE DELETION PROCESSING IN THE CONTEXT OF THE TARGET
226 ;       PROCESS.  A PROCESS THEREFORE DELETES ITSELF.
227 ;
228 ; CALLING SEQUENCE:
229 ;       CALLG   ARGLIST,G^EXE$DELPRC
230 ;
231 ; INPUT PARAMETERS:
232 ;       PIDADR(AP)  - ADDRESS OF THE PROCESS IDENTIFICATION OF THE
233 ;                     PROCESS TO BE DELETED.
234 ;       PRCNAM(AP) - ADDRESS OF STRING DESCRIPTOR FOR PROCESS LOGICAL NAME
235 ;                     STRING.
236 ;       R4 - PCB ADDRESS OF CURRENT PROCESS
237 ;
238 ; IMPLICIT INPUTS:
239 ;       CURRENT PROCESS PCB AND PHD
240 ;       PCB OF TARGET PROCESS
241 ;
242 ; OUTPUT PARAMETERS:
243 ;       R0 - COMPLETION STATUS CODE
244 ;       @PIDADR(AP) - PROCESS IDENTIFICATION OF DELETED PROCESS
245 ;
246 ; IMPLICIT OUTPUTS:
247 ;       NONE
248 ;
249 ; COMPLETION CODES:
250 ;       SS$_NORMAL    -   SUCCESSFUL COMPLETION
251 ;       SS$_NONEXPR   -   NONEXISTENT PROCESS
252 ;       SS$_NOPRIV    -   NO PRIVILEGE, TARGET PROCESS IS NOT A SUBPROCESS
253 ;                         AND ISSUING PROCESS DOES NOT HAVE PROCESS CONTROL
254 ;                         PRIVILEGE FOR GROUP OR WORLD.
255 ;       SS$_INSFMEM   -   INSUFFICIENT DYNAMIC MEMORY AVAILABLE FOR SERVICE
256 ;                         AND RESOURCE WAIT MODE DISABLED.
257 ;
258 ; SIDE EFFECTS:
259 ;       A KERNEL AST WILL BE ENQUEUED FOR THE PROCESS TO BE
260 ;       DELETED WHICH MAY CAUSE RESCHEDULING TO OCCUR.
261 ;
262 ;--
263
264          SYSTEM_SERVICE  DELPRC,-
265                          <R2,R3,R4,R5,R6,R7>,-
266                          MODE=KERNEL,-
267                          NARG=2
268
269          CLRL    R5                      ; Clear pointer to ACB
270          MOVL    R4,R6                   ; Save current process' PCB address
271          BSBW    EXE$NAMPID              ; Convert name/PID to PCB address
272          BLBC    R0,130$                 ; Exit if error
273
274 ; R4 -> PCB of target process
275 ; R5 =  0
```

```
276 ; R6 -> PCB of current process
277 ; IPL = SYNCH
278
279         BBS     #PCB$V_NODELET,-        ; See if process can be deleted
280                 PCB$L_STS(R4),150$
281
282 ; Allocate an AST control block to send to the process
283
284         MOVZBL  #ACB$K_LENGTH,R1        ; Get an ACB to send away
285         JSB     G^EXE$ALONONPAGED       ; Allocate AST control block
286         BLBC    R0,110$                 ; Return error if none
287         ASSUME  ACB$B_RMOD EQ ACB$B_TYPE+1
288         MOVZBW  #DYN$C_ACB,-            ; Set type, zero RMOD field
289                 ACB$B_TYPE(R2)
290         MOVW    R1,ACB$W_SIZE(R2)       ; Set actual size allocated
291         MOVL    R2,R5                   ; Set address for SCH$QAST
292
293 ; R4 -> PCB of target process
294 ; R5 -> ACB to deliver to target
295 ; R6 -> PCB of current process
296 ; IPL = SYNCH
297
298         BBSS    #PCB$V_DELPEN,-         ; BR if target process is already
299                 PCB$L_STS(R4),140$      ; marked for deletion
300
301 ; The following three instructions are the guts of the $RESUME system service.
302 ; That same functionality must be reproduced here while remaining at IPL
303 ; SYNCH so the system service cannot be used.
304
305         ASSUME PCB$V_RESPEN LE 7
306
307 100$:   BISB    #<1@PCB$V_RESPEN>,PCB$L_STS(R4) ; Set resume pending bit
308         MOVZBL  #PRI$_RESAVL,R2         ; Set priority increment class
309         RPTEVT  RESUME                  ; Report RESUME event to scheduler
310
311         MOVAB   W^DELETE,ACB$L_AST(R5)  ; Set address for delete action
312         CLRB    ACB$B_RMOD(R5)          ; Mark as normal kernel mode AST
313         MOVL    PCB$L_PID(R4),ACB$L_PID(R5)     ; Set PID of target
314
315 ; R2 has already been loaded with the correct priority increment class
316 ; and R2 is preserved by G^SCH$RSE (called from within the RPTEVT macro)
317 ; so that a
318 ;
319 ;       MOVZBL  #PRI$_RESAVL,R2         ; Set priority increment
320 ;
321 ; is not necessary here.
322
323         JSB     G^SCH$QAST              ; And queue ast for process
324         MOVZWL  #SS$_NORMAL,R0          ; Set normal status
325
326 110$:   UNLOCK  LOCKNAME=SCHED,-        ; Unlock SCHED database
327                 PRESERVE=YES            ; Don't preserve R0
328                                         ;  SCHED was locked by EXE$NAMPID
329
330 120$:   SETIPL  #0                      ; Drop IPL lower
331         RET                             ; ... and return
332
```

```
  333 ; Error from EXE$NAMPID.  Check to see if it is a remote call.  If not,
  334 ; lower IPL and return with the error in R0.
  335
  336 130$:   CMPW    #SS$_REMOTE_PROC,R0     ; Is it a call to a remote process?
  337         BNEQ    120$                    ; Not remote, continue with error
  338
  339         ASSUME DYN$C_CWPS_DELPRC     EQ   CWPSSRV$K_DELPRC
  340         MOVZBL  #DYN$C_CWPS_DELPRC,R3   ; Load subtype code for CWPS
  341         BRW     CWPS$PCNTRL             ; Branch to the cluster code
  342
  343 ; The target process is already marked for deletion.
  344 ;
  345 ; If the target process is not the current process, the system service
  346 ; simply returns a success code to the caller after deallocating the ACB.
  347 ; Note that SS$_NORMAL was loaded into R0 by the success path of EXE$NAMPID.
  348 ;
  349 ; If, however, the target process is the current process, and exec mode
  350 ; rundown is active, allow deletion to continue. This is done to guarantee
  351 ; that a process that issues a $EXIT or $DELPRC call during exec mode rundown
  352 ; processing will still be deleted.
  353 ;
  354 ; Note that the ERDACT bit remains set in the PCB status longword. This prevents
  355 ; another inititation of exec mode rundown processing.
  356
  357 140$:   CMPL    R4,R6                   ; Current process being deleted ?
  358         BNEQ    160$                    ; No, deallocate ACB
  359         BBC     #PCB$V_ERDACT,-         ; BR if exec mode rundown not active
  360                 PCB$L_STS(R4),160$
  361         BRB     100$                    ; Otherwise, continue with deletion
  362
  363 ; The following error paths must deallocate the unused ACB before returning.
  364 ;
  365 150$:   MOVZWL  #SS$_NODELETE,R0        ; Process cannot be deleted
  366 160$:   PUSHL   R0                      ; Save status across next two calls
  367         UNLOCK  LOCKNAME=SCHED,-        ; Unlock SCHED database
  368                 NEWIPL=#IPL$_ASTDEL,-   ; Drop IPL, but not below ASTDEL
  369                 PRESERVE=NO             ;  SCHED was locked by EXE$NAMPID
  370
  371         MOVL    R5,R0                   ; Get address of ACB
  372         BEQL    170$                    ; Not yet allocated, don't delete
  373         JSB     G^EXE$DEANONPAGED       ; Deallocate packet back to pool
  374 170$:   POPL    R0                      ; Restore status
  375         BRB     120$                    ; Join common exit
  376
```

```
; The rest of the code in this module can exist in a pageable part of the exec

        DECLARE_PSECT   EXEC$PAGED_CODE
        .SBTTL  DELETE - PERFORM DELETE ACTIONS IN CONTEXT OF PROCESS
;++
; FUNCTIONAL DESCRIPTION: DELETE EXECUTES AS THE RESULT OF A KERNEL
;       AST INITIATED BY THE DELETE PROCESS SYSTEM SERVICE.  IT PERFORMS
;       THE ACTIONS NECESSARY TO DELETE A PROCESS AND RETURN ITS RESOURCES.
;       TERMINATION MESSAGES ARE SENT TO THE TERMINATION MAILBOX UNIT
;       SPECIFIED WHEN THE PROCESS WAS CREATED AND TO THE SYMBIONT
;       MANAGER IF ACCOUNTING IS NOT DISABLED FOR THIS PROCESS.
;
; CALLING SEQUENCE:
;       (SAME EFFECT AS) DCLAST ASTADR=DELETE MODE=KERNEL
;
; INPUT PARAMETERS:
;       NONE
;
; OUTPUT PARAMETERS:
;       NONE
;
; IMPLICIT INPUTS:
;       PCB OF CURRENT PROCESS (LOCATED VIA CTL$GL_PCB)
;       PHD OF CURRENT PROCESS
;
; IMPLICIT OUTPUTS:
;       NONE
;
; SIDE EFFECTS:
;       NONE
;--

; Note: No registers need to be saved, nor does the exec stack need to be
;       cleaned, since control will never return to the interrupted thread
;       of execution.

EXEC_RUNDOWN_AST:                       ; Call exec mode rundown handlers
        .WORD
        MOVL    G^CTL$GL_PCB,R4         ; Get PCB address
        CALLS   #0,G^SYS$CLRAST         ; Return to exec mode, non-AST level
        ADDL2   S^#<EXE$C_CMSTKSZ+<6*4>>,SP ; Cleanup AST delivery stack
        MOVL    #PSL$C_EXEC,R7          ; Exec mode rundown
        PUSHL   AP                      ; Save current AP
        PUSHL   #PSL$C_EXEC             ; Another exec mode access parameter
        PUSHL   #1
        MOVL    SP,AP
        MOVL    G^CTL$GL_USRUNDWN_EXEC,R5 ; Fetch address of (first) per-process han
        BEQL    100$                    ; BR if none
        JSB     (R5)                    ; Call thru the vector(s)
100$:   MOVL    G^EXE$GL_USRUNDWN_EXEC,R5 ; Fetch address of (first) system-wide han
        BEQL    110$                    ; BR if none
        JSB     (R5)                    ; Call thru the vector(s)
110$:   ADDL    #8,SP                   ; Remove stack parameters
        MOVL    (SP)+,AP                ; and restore AP
        $CMKRNL_S ROUTIN=B^DELETE_CONTINUE
120$:   BRB     120$
;       BUG_CHECK xxxx,FATAL            ; Should not return here !!!
```

```
435
436 DELETE_CONTINUE:                         ; Proceed with process deletion
437         .WORD                           ; (R4 contains PCB from CHMK dispatcher)
438         ADDL2   S^#<EXE$C_CMSTKSZ+EXE$C_CMSTKSZ+<2*4>>,SP; Cleanup kernel stack
439         BRB     DELETE_2                ; Join main-line delete process code
440                                         ; Note: PCB$V_ERDACT remains set to
441                                         ;       insure that exec mode rundown
442                                         ;       will not be re-initiated because
443                                         ;       of an exception or bugcheck, or
444                                         ;       due to calls to $EXIT or $DELPRC
445 EXEC_RUNDOWN:
446         SETIPL  #IPL$_ASTDEL            ; Protect against kernel mode ASTs
447         MOVQ    R2,-(SP)                ; Save working registers
448         MOVL    #ACB$C_LENGTH,R1        ; Allocate ACB
449         JSB     G^EXE$ALLOCBUF          ; Should return only upon success
450         BLBC    R0,10$
451                                         ; Fill in ACB
452         MOVB    #DYN$C_ACB,ACB$B_TYPE(R2); Dynamic structure type
453         MOVB    #PSL$C_EXEC,ACB$B_RMOD(R2); Exec mode, QUOTA/NODELETE bits clear
454         MOVL    PCB$L_PID(R4),ACB$L_PID(R2); PID of target process
455         MOVAB   W^EXEC_RUNDOWN_AST,ACB$L_AST(R2); Address of AST routine
456         ASSUME  ACB$L_KAST EQ ACB$L_ASTPRM+4
457         CLRQ    ACB$L_ASTPRM(R2)        ; No AST param or special KAST routine
458         MOVL    R2,R5                   ; ACB address must be R5
459         MOVZBL  #PRI$_RESAVL,R2         ; Priority increment class
460         JSB     G^SCH$QAST              ; Queue the AST
461         MOVQ    (SP)+,R2                ; Restore working registers
462         SETIPL  #0                      ; Drop IPL and
463         RET                             ; dismiss process deletion AST (for now)
464
465 ;10$:   BUG_CHECK xxxx,FATAL            ; ALLOCBUF returned with failure !!!
466 10$:    MOVQ    (SP)+,R2                ; ALLOCBUF returned with failure !!!
467         SETIPL  #0
468         BRB     DELETE_1
469
470         .ALIGN  LONG
471 DELETE:                                 ; PERFORM DELETE OPERATIONS
472         .WORD   ^M<R4,R5>               ; ENTRY MASK
473         MOVL    G^CTL$GL_PCB,R4         ; GET PCB ADDRESS
474         BBCC    #PCB$V_SSRWAIT,PCB$L_STS(R4),100$ ; ENABLE RESOURCE WAIT
475 100$:   MOVL    G^CTL$GL_USRUNDWN_EXEC,R5; Per-process exec mode rundown handlers de
476         BEQL    110$                    ; BR if no
477         CMPB    #OP$_RSB,(R5)           ; If JSB target a 'RSB' instruction ?
478         BNEQ    120$                    ; BR if no
479 110$:   MOVL    G^EXE$GL_USRUNDWN_EXEC,R5; System-wide exec mode rundown handlers de
480         BEQL    130$                    ; BR if no
481         CMPB    #OP$_RSB,(R5)           ; If JSB target a 'RSB' instruction ?
482         BEQL    130$                    ; BR if yes
483
484 120$:   BBCS    #PCB$V_ERDACT,-         ; BR if exec mode rundown is not active
485                 PCB$L_STS(R4),EXEC_RUNDOWN
486
487 130$:
488 DELETE_1:
489         BICB2   #1, PCB$B_ASTACT(R4)    ; CLEAR KERNEL AST ACTIVE
490         JSB     G^SCH$NEWLVL            ; COMPUTE NEW AST LEVEL
491 DELETE_2:
```

```
SYSDELPRC - DELETE PROCESS SYSTEM SERVICE 10-MAY-1989 16:40:14 VAX MACRO V5.0-8 Page 11
X-33 DELETE - PERFORM DELETE ACTIONS IN CONTE 20-DEC-1988 12:59:50 [SYS.SRC]SYSDELPRC.MAR;

  492          BSBW    DELETE_WAIT             ; WAIT FOR DPC
  493
  494          .SAVE_PSECT
  495          DECLARE_PSECT   EXEC$NONPAGED_CODE
  496
  497 DELETE_WAIT:
  498          LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
  499                  PRESERVE=NO             ; OK to destroy R0
  500          TSTB    PCB$B_DPC(R4)           ; IS DPC NON-ZERO?
  501          BEQL    100$                    ; EQL THEN DELETION IS ALLOWED
  502          PUSHL   #0                      ; KERNEL MODE, IPL 0 PSL FOR RWAIT
  503          MOVL    #RSN$_ASTWAIT, R0       ; NOTE AST RESOURCE
  504          JSB     G^SCH$RWAIT             ; WAIT FOR AST
  505          BRB     DELETE_WAIT             ; MAKE THE TEST AGAIN
  506 100$:    UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATABASE
  507                  NEWIPL=#0,-             ; DROP IPL
  508                  PRESERVE=NO             ; OK to destroy R0
  509          RSB
  510
  511          .RESTORE_PSECT
  512 ;-------------------------------------------------------------
  513 ;
  514 ;        INVOKE THE USER RUNDOWN SERVICE(S)
  515 ;
  516 ;        The user service is invoked in KERNEL mode with a JSB and must
  517 ;        exit with a RSB.
  518 ;
  519 ;        R4 - Pointer to current PCB
  520 ;        R7 - Acess mode of call to rundown routine (always 0)
  521 ;        ACMODE(AP) - always 0
  522 ;
  523 ;        The above parameters are the same as in the call from SYSRUNDWN.
  524 ;
  525 ;-------------------------------------------------------------
  526
  527          CLRL    R7                      ; KERNEL MODE RUNDOWN
  528          PUSHL   AP
  529          CLRL    -(SP)                   ; ANOTHER KERNEL MODE ACCESS PARAMETER
  530          PUSHL   #1
  531          MOVL    SP,AP
  532          MOVL    G^CTL$GL_USRUNDWN, R5   ; GET PER-PROCESS USER RUNDOWN VECTOR
  533          BEQL    110$                    ; NOT PRESENT, SKIP ON
  534          JSB     (R5)                    ; CALL THRU THE VECTOR(S)
  535 110$:    MOVL    G^EXE$GL_USRUNDWN, R5   ; GET SYSTEM-WIDE USER RUNDOWN VECTOR
  536          BEQL    120$                    ; NOT PRESENT, SKIP ON
  537          JSB     (R5)                    ; CALL THRU THE VECTOR(S)
  538 120$:    ADDL    #8,SP                   ; CLEAN UP
  539          MOVL    (SP)+,AP
  540          JSB     G^EXE$RESETVEC1         ; RESET THE PRIVILEGED LIBRARY VECTORS
  541
  542 ;-------------------------------------------------------------
  543 ;
  544 ;        RMS RUNDOWN
  545 ;
  546 ;-------------------------------------------------------------
  547
  548          MOVAB   -136(SP),SP             ; ALLOCATE STRING BUFFER
```

```
549 130$:   MOVZBL  #128,(SP)               ; SET COUNT FOR STRING BUFFER
550         MOVAL   8(SP),4(SP)             ; FORM ADDRESS PART OF DESCRIPTOR
551         PUSHL   #2                      ; LAST CHANCE RMS CLEANUP
552         PUSHAB  4(SP)                   ; SET ADDRESS OF STRING BUFFER
553         CALLS   #2,G^SYS$RMSRUNDWN      ; DEFAULT CLOSE OF OPEN RMS FILES
554         BLBC    R0,130$                 ; REPEAT IF MORE TO CLOSE
555
556 ;-----------------------------------------------------------------------
557 ;
558 ;       FIND AND DELETE ALL SUB-PROCESSES
559 ;
560 ;-----------------------------------------------------------------------
561         .ENABLE LOCAL_BLOCK
562 SUBDELETE:                              ; DELETE SUB-PROCESSES
563         TSTW    PCB$W_PRCCNT(R4)        ; ANY SUB-PROCESSES?
564         BEQL    50$                     ; NO, CONTINUE
565         MOVZWL  G^SCH$GL_MAXPIX,R6      ; INITIALIZE INDEX FOR PROCESS SCAN
566 10$:    MOVL    @W^SCH$GL_PCBVEC[R6],R3 ; GET A PCB ADDRESS
567         CMPL    PCB$L_PID(R4),PCB$L_OWNER(R3)   ; DO WE OWN IT
568         BNEQ    20$                     ; NO, TRY ANOTHER
569         $DELPRC_S       PCB$L_EPID(R3)  ; MARK IT FOR DELETE USING THE EXTENDED PID
570 20$:    SOBGTR  R6,10$                  ; CONTINUE
571         BSBW    WAIT_FOR_SUBPRC         ; Wait for sub-processes
572
573         .SAVE_PSECT
574         DECLARE_PSECT   EXEC$NONPAGED_CODE
575
576 WAIT_FOR_SUBPRC:
577         SAVIPL  R6                      ; SAVE IPL
578 30$:    MOVPSL  -(SP)                   ; SAVE PSL
579         LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
580                 PRESERVE=NO             ; DON'T PRESERVE R0
581         TSTW    PCB$W_PRCCNT(R4)        ; CHECK COUNT OF SUB-PROCESSES STILL RUNNING
582         BEQL    40$                     ; DONE
583         MOVZWL  #RSN$_ASTWAIT,R0        ; SET RESOURCE NUMBER
584         JSB     G^SCH$RWAIT             ; AND WAIT FOR AN AST
585         BRB     30$                     ; CHECK AGAIN WHEN AWAKENED
586 40$:    UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATABASE
587                 NEWIPL=R6,-             ; RESTORE PREVIOUS IPL
588                 PRESERVE=NO             ; DON'T PRESERVE R0
589         TSTL    (SP)+                   ; Clean off stack
590         RSB
591
592         .RESTORE_PSECT
593 50$:                                    ; DONE, ALL SUB-PROCESSES DELETED
594
595         .DISABLE LOCAL_BLOCK
596
597 ;-----------------------------------------------------------------------
598 ;
599 ;       RUNDOWN PROCESS ACTIVITY
600 ;
601 ;-----------------------------------------------------------------------
602
603 RUNDWN:                                 ; RUNDOWN PROCESS ACTIVITY
604         PUSHL   #0                      ; SET MODE FOR RUNDOWN
605         CALLS   #1,G^SYS$RUNDWN         ; RUNDOWN AT KERNEL MODE
```

```
606          MOVL    G^CTL$GL_PHD,R5          ; GET PROCESS HEADER ADDRESS
607          ADDL3   PHD$L_PSTBASOFF(R5),R5,R2 ; GET ADDRESS OF SECTION TABLE
608          CLRL    -(SP)                    ; ASSUME NO PAGES WILL BE DELETED
609          SUBL    #16,SP                   ; SAVE ROOM FOR INRANGE AND RETRANGE
610          MOVL    SP,R3                    ; REMEMBER SCRATCH AREA ADDRESS
611          CVTWL   PHD$W_PSTLAST(R5),R6     ; GET INDEX TO LAST SEC TBL ENTRY USED
612          BRB     140$                     ; JOIN COMMON LOOP
613 100$:    TSTL    SEC$L_REFCNT(R2)[R6]     ; ANY PAGES MAPPED?
614          BLSS    130$                     ; BR IF NOTHING MAPPED TO THIS SECTION
615          MOVAL   (R2)[R6],R7              ; ADDRESS OF SECTION TABLE ENTRY
616          BBCC    #SEC$V_INPROG,SEC$W_FLAGS(R7),110$ ; BR if section creation not in p
617          MOVL    R6,R1                    ; R1 = PSTX
618          PUSHL   R2                       ; Preserve volatile register
619          JSB     G^MMG$DECSECREFL         ; Remove bias from reference count
620          POPL    R2                       ; Restore PST base address
621          TSTL    SEC$L_REFCNT(R7)         ; Did ref count just go to zero ?
622          BNEQ    110$                     ; BR if no
623          MOVL    #1,16(R3)                ; Indicate second rundown required (to deass
624 ;
625 ; DO A $DELTVA FOR THIS SECTION.
626 ;
627 110$:    EXTZV   #SEC$V_VPX,#SEC$S_VPX,SEC$L_VPXPFC(R7),4(R3) ; GET START VPN
628          CMPZV   #PHD$V_P0LR,#PHD$S_P0LR,PHD$L_P0LRASTL(R5),4(R3) ; IS VA IN P0?
629          BGTRU   120$                     ; BR IF IT IS IN P0 SPACE
630          ROTL    #7,G^SGN$GL_PTPAGCNT,R0  ; GET LONGWD OFFSET FROM BEGIN OF PAGTBL
631          SUBL    R0,4(R3)                 ; GET VPN IN P1 SPACE
632          EXTZV   #SEC$V_VPX,#SEC$S_VPX,4(R3),4(R3) ; MASK OFF SIGN BITS
633 120$:    ASHL    #9,4(R3),4(R3)           ; GET VA OF FIRST PAGE MAPPED
634          ASHL    #9,SEC$L_PAGCNT(R7),(R3) ; GET # OF BYTES IN SECTION
635          ADDL2   4(R3),(R3)               ; GET VA OF LAST PAGE MAPPED
636          DECL    (R3)                     ; GET ADR OF LAST BYTE MAPPED
637          $DELTVA_S       (R3),8(R3)       ; DELETE PAGES FOR THIS SECTION
638          CMPL    #-1,8(R3)                ; WERE ANY PAGES DELETED?
639          BEQL    130$                     ; BR IF NONE DELETED
640          MOVL    #1,16(R3)                ; INDICATE SECOND RUNDWN REQUIRED
641 130$:    ADDL2   #SEC$C_LENGTH@-2,R6      ; GET OFFSET TO NEXT SECTION TABLE ENTRY
642 140$:    BLSS    100$                     ; GO CHECK NEXT SECTION
643          ADDL    #16,SP                   ; CLEAN INRANGE AND RETRANGE OFF STACK
644          TSTL    (SP)+                    ; WERE ANY PAGES DELETED?
645          BEQL    180$                     ; NO, THEN DON'T BOTHER WITH 2ND RUNDWN
646          SETIPL  #IPL$_ASTDEL
647          JSB     G^MMG$DALCSTXSCN         ; Deallocate any unreferenced PSTs
648          SETIPL  #0
649          PUSHL   #0                       ; SET MODE FOR RUNDOWN
650          CALLS   #1,G^SYS$RUNDWN          ; RUNDOWN AT KERNEL MODE
651
652 ;*****************************************************************
653 ;
654 ; SCAN THE CHANNEL TABLE AND LOOK FOR ASSIGNED
655 ; CHANNELS. THERE SHOULD BE NONE.
656 ;
657 ;*****************************************************************
658
659 150$:    MOVZWL  G^CTL$GW_CHINDX,R5       ; GET MAXIMUM INDEX + 16
660          BEQL    180$                     ; NO CHANNEL TABLE
661          MNEGL   R5,R5                    ; CONVERT TO NEGATIVE OFFSET
662          ADDL3   #CCB$B_AMOD, -
```

```
                        G^CTL$GL_CCBBASE,R3     ; COMPUTE ADDRESS OF HIGHEST CHANNEL
160$:   TSTB    (R3)[R5]                ; IS THIS CHANNEL STILL ASSIGNED?
        BLEQ    170$                    ; BRANCH IF NOT.

;
; A NEGATIVE ACCESS MODE IN THE CCB IS USED TO RESERVE THE CHANNEL
; FOR USE BY THE F11BXQP TO PERFORM LOGICAL I/O ON BEHALF OF VIRTUAL
; I/O FUNCTIONS THAT IT PROCESSES (THE XQP MODIFIES THE CCB$L_UCB AND
; CCB$B_AMOD FIELDS PRIOR TO ACTUALLY USING IT).  THE CHANNEL DOES NOT
; HAVE TO BE DEASSIGNED BECAUSE NO DEVICE REFERENCE COUNTS REFLECT ITS
; EXISTENCE.
;

        BUG_CHECK FILCNTNONZ,FATAL      ; IT REALLY SHOULD HAVE WORKED

170$:   ADDL2   #CCB$C_LENGTH,R5        ; NEXT CHANNEL
        BLSS    160$                    ; LOOP THRU
180$:

;*****************************************************************************
; END OF BUG TRAP
;*****************************************************************************

;-----------------------------------------------------------------------------
;
;       DISMOUNT ALL MOUNTED VOLUMES
;
;-----------------------------------------------------------------------------

        MOVL    PCB$L_JIB(R4),R3        ; GET JIB ADDRESS
        BNEQ    190$                    ; IF NEQ, GOT A JIB
        BRW     270$                    ; ELSE, CONTINUE

190$:   TSTL    PCB$L_OWNER(R4)         ; IS THIS A SUBPROCESS ?
        BNEQ    200$                    ; IF NEQ YES
;
;       MAIN PROCESS - DISMOUNT VOLUME
;
        JSB     G^SCH$IOLOCKW           ; LOCK I/O DATABASE
        REMQUE  @JIB$L_MTLFL(R3),R6     ; GET A VOLUME TO BE DISMOUNTED
        BVS     260$                    ; NONE, FINISHED WITH DISMOUNT LIST
        JSB     G^SCH$IOUNLOCK          ; UNLOCK I/O DATABASE
        SETIPL  #0                      ; DROP IPL
        CLRL    R3                      ; UNLOAD UPON DISMOUNT
        JSB     G^IOC$DISMOUNT          ; DISMOUNT IT
        BRW     150$                    ; AND TRY FOR ANOTHER
;
;       SUBPROCESS - MAKE JOB MASTER THE DEVICE OWNER IF DEVICE IS STILL
;                    ALLOCATED TO THIS SUBPROCESS. SET THE DEADMO BIT SUCH
;                    THAT THE DEVICE WILL BE DEALLOCATED ON DISMOUNT (SINCE
;                    THE ORIGINAL ALLOCATOR HAS BEEN DELETED).
;
200$:   JSB     G^SCH$IOLOCKW           ; LOCK I/O DATABASE
        MOVL    JIB$L_MTLFL(R3),R6      ; GET ADDRESS OF FIRST MTL
210$:   CMPL    R3,R6                   ; END OF MTL LIST ?
        BEQL    260$                    ; IF EQL YES
        MOVL    MTL$L_UCB(R6),R5        ; GET UCB ADDRESS
```

```
SYSDELPRC - DELETE PROCESS SYSTEM SERVICE 10-MAY-1989 16:40:14 VAX MACRO V5.0-8 Page 15
X-33 DELETE - PERFORM DELETE ACTIONS IN CONTE 20-DEC-1988 12:59:50 [SYS.SRC]SYSDELPRC.MAR;1

  720          CMPL    UCB$L_PID(R5),PCB$L_PID(R4) ; DEVICE ALLOCATED TO THIS SUBPROCESS?
  721          BNEQ    220$                        ; IF NE NO, SKIP OWNERSHIP TRANSFER
  722          MOVL    JIB$L_MPID(R3),UCB$L_PID(R5) ; MAKE JOB MASTER THE DEVICE OWNER
  723          BISW    #UCB$M_DEADMO,UCB$W_STS(R5)  ; DEALLOCATE DEVICE ON DISMOUNT
  724
  725 220$:    BBS     #DEV$V_SQD,UCB$L_DEVCHAR(R5),-
  726                  240$                        ; SKIP IF SEQUENTIAL DEVICE
  727          BBC     #DEV$V_MSCP,UCB$L_DEVCHAR2(R5),-
  728                  240$                        ; SKIP IF NON-MSCP
  729          TSTW    UCB$W_MSCPUNIT(R5)          ; IS THIS A SHADOW SET VIRTUAL UNIT ?
  730          BGEQ    240$                        ; IF GEQ NO, SKIP SHADOW MEMBER PROCESSING
  731
  732          ASSUME  <VCB$B_STATUS-VCB$L_MEMQFL> EQ <VCB$B_SHAD_STS-VCB$L_MEMHDFL>
  733          ASSUME  VCB$V_SHADMAST EQ 0
  734
  735          MOVL    UCB$L_VCB(R5),R0            ; GET VIRTUAL UNIT VCB ADDRESS
  736          MOVAB   <VCB$L_MEMHDFL -            ; INITIALIZE MEMBERS QUEUE
  737                  -VCB$L_MEMQFL>(R0),R0       ;   ADDRESS
  738 230$:    MOVL    VCB$L_MEMQFL(R0),R0         ; GET NEXT MEMBER VCB
  739          BLBS    VCB$B_STATUS(R0),240$       ; IF LBS, END OF MEMBER VCB LIST
  740          MOVL    VCB$L_MEM_UCB(R0),R5        ; GET MEMBER UCB ADDRESS
  741          CMPL    UCB$L_PID(R5),PCB$L_PID(R4) ; DEVICE ALLOCATED TO THIS SUBPROCESS?
  742          BNEQ    230$                        ; IF NE NO, SKIP OWNERSHIP TRANSFER
  743          MOVL    JIB$L_MPID(R3),UCB$L_PID(R5) ; MAKE JOB MASTER THE DEVICE OWNER
  744          BISW    #UCB$M_DEADMO,UCB$W_STS(R5)  ; DEALLOCATE DEVICE ON DISMOUNT
  745          BRB     230$                        ; GO PROCESS NEXT SHADOW SET MEMBER
  746
  747 240$:    MOVL    MTL$L_MTLFL(R6),R6          ; GET NEXT MTL ENTRY
  748          BRW     210$                        ; AND TRY FOR ANOTHER
  749
  750 250$:    BUG_CHECK  INCONSTATE               ; DIOCNT should not be > DIOLM
  751          BRB     280$                        ; Continue after writing error log
  752
  753 260$:    JSB     G^SCH$IOUNLOCK              ; UNLOCK I/O DATABASE
  754          SETIPL  #0                          ; DROP IPL
  755
  756 270$:    $DALLOC_S       ACMODE=#0           ; DEALLOCATE ALL
  757
  758          CMPW    PCB$W_DIOCNT(R4),PCB$W_DIOLM(R4) ; WAIT FOR DIRECT I/O COMPLETION
  759          BLSS    270$                        ; IF < OK
  760          BGTR    250$                        ; IF > CRASH
  761 280$:    CMPW    PCB$W_BIOCNT(R4),PCB$W_BIOLM(R4) ; AND BUFFERED I/O
  762          BNEQ    280$                        ;
  763
  764 ;_____________________________________________________________________
  765 ;
  766 ;        DECREMENT INTERACTIVE AND BATCH JOB COUNT WHEN APPROPRIATE
  767 ;
  768 ;---------------------------------------------------------------------
  769
  770          TSTL    PCB$L_OWNER(R4)
  771          BNEQ    RELQUOTA                    ; WE DON'T DO IT FOR A SUBPROCESS
  772          BBC     #PCB$V_INTER,PCB$L_STS(R4),290$ ; NOT INTERACTIVE
  773          DECW    G^SYS$GW_IJOBCNT            ; ONE LESS INTERACTIVE JOB
  774          BRB     DELETE_JT
  775 290$:    BBC     #PCB$V_BATCH,-              ; NOT BATCH JOB
  776                  PCB$L_STS(R4),DELETE_JT
```

```
777         DECW    G^SYS$GW_BJOBCNT                ; ONE LESS BATCH JOB
778
779 ;-----------------------------------------------------------------------
780 ;
781 ;       DELETE THE JOB-WIDE LOGICAL NAME TABLE IF THE PROCESS
782 ;       IS NOT A SUB-PROCESS
783 ;
784 ;-----------------------------------------------------------------------
785
786 DELETE_JT:                                 ; DELETE THE JOB-WIDE LOGICAL NAME TABLE
787         MOVAB   -JOB_STRING_LEN(SP),SP      ; MAKE ROOM ON STACK FOR TABLE NAME
788         PUSHAB  (SP)                        ; CONSTRUCT A DESCRIPTOR OF THE BUFFER
789         PUSHL   #JOB_STRING_LEN             ; WHICH IS TO CONTAIN THE TABLE NAME
790         MOVL    SP,R3                       ; SAVE THE ADDRESS OF THE DESCRIPTOR
791
792         $FAO_S  -                           ; CONSTRUCT THE JOB-WIDE TABLE NAME
793                 CTRSTR  = JOB_FAO, -
794                 OUTBUF  = (R3), -
795                 P1      = PCB$L_JIB(R4)
796
797         $DELLNM_S       -                   ; DELETE THE JOB-WIDE LOGICAL NAME TABLE
798                 LOGNAM  = (R3), -
799                 TABNAM  = LNM_SYSTEM_DIRECTORY_DESC
800         ADDL2   #JOB_STRING_LEN+8,SP        ; RESTORE THE STACK
801
802 ;-----------------------------------------------------------------------
803 ;
804 ;       SEND REMAINING QUOTAS TO OWNER IF SUB-PROCESS
805 ;
806 ;-----------------------------------------------------------------------
807 RELQUOTA:                                   ; RELEASE QUOTAS
808         CLRB    PCB$T_LNAME(R4)             ; REMOVE PROCESS NAME
809         MOVL    PCB$L_OWNER(R4),R6          ; GET PID OF OWNER
810         BEQL    MESSAGES                    ; IF EQL, NOT A SUBPROCESS
811         PUSHL   R4                          ; SAVE PCB ADDRESS
812         MOVZWL  R6,R6                       ; EXTRACT PIX FROM OWNER PID
813         MOVL    @W^SCH$GL_PCBVEC[R6],R4     ; AND GET PCB
814         JSB     G^EXE$ALLOCIRP              ; ALLOCATE AST CONTROL BLOCK
815         BLBC    R0,20$                      ; SKIP QUOTA RETURN IF ERROR
816         MOVL    R2,R5                       ; SET ADDRESS OF QUOTA AST BLOCK
817         MOVAB   W^RETQUOTA,ACB$L_KAST(R5)       ; SET AST ADDRESS FOR SPECIAL
818                                             ; KERNEL AST TO RETURN QUOTAS
819         MOVB    #<1@ACB$V_KAST>,ACB$B_RMOD(R5)  ; FLAG AS SPECIAL KERNEL AST
820         MOVL    PCB$L_PID(R4),ACB$L_PID(R5)     ; SET TARGET PID FOR AST
821         MOVL    (SP),R2                     ; GET PCB ADDRESS FOR PROCESS BEING DELETED
822         MOVL    G^CTL$GL_PHD,R1             ; GET PROCESS HEADER ADDRESS
823         SUBL3   PHD$L_EXTRACPU(R1),PHD$L_CPULIM(R1),ACB_L_CPULIM(R5)
824                                             ; SAVE CPULIMIT LESS BONUS
825         MOVL    PHD$L_CPUTIM(R1),ACB_L_CPUTIM(R5)
826                                             ;  AND ACCUMULATED CPU TIME
827         MOVZWL  #PRI$_RESAVL,R2             ; SET PRIORITY INCREMENT CLASS
828         JSB     G^SCH$QAST                  ; QUEUE AST FOR CREATOR
829 20$:    POPR    #^M<R4>                     ; RESTORE PCB ADDRESS
830         SETIPL  #0                          ; DROP IPL TO ISSUE SYSTEM SERVICE REQUESTS
831
832 ;-----------------------------------------------------------------------
833 ;
```

```
834 ;       TERMINATION MESSAGES
835 ;
836 ;---------------------------------------------------------------------
837 MESSAGES:                                ; SEND TERMINATION MESSAGES
838         TSTW    PCB$W_TMBU(R4)          ; TERMINATION MAILBOX SPECIFIED ?
839         BEQL    10$                     ; IF EQL, NO
840         BSBW    TERMMSG                 ; SEND MESSAGE TO TERMINATION MAILBOX
841 10$:                                    ; SEND MESSAGE TO JOB CONTROLLER
842         BBC     #PCB$V_BATCH,PCB$L_STS(R4),20$  ; IF BC, NOT A BATCH JOB
843         BBCC    #PCB$V_SSRWAIT,PCB$L_STS(R4),20$; ENABLE RESOURCE WAIT MODE
844 20$:
845         CLRL    R5                      ; NOT A SPECIAL KERNEL AST ROUTINE
846         JSB     G^EXE$PRCDELMSG         ; SEND PROCESS DELETE MESSAGE
847
848 ;---------------------------------------------------------------------
849 ;
850 ;       DESTROY PROCESS
851 ;
852 ;---------------------------------------------------------------------
853
854
855 DESTROY:                                ;
856         SETIPL  #IPL$_ASTDEL            ; NO MORE ASTS AFTER THIS POINT
857         SUBL    #16,SP                  ; 4 LONGWORDS OF STORAGE
858         MOVL    SP,R2                   ; SAVE BASE
859         MOVL    #<<1@31>-1>,(R2)        ; SET STARTVA
860         MOVL    #<1@30>,4(R2)           ; END AT VECTOR PAGE
861                                         ; DELTVA GOES IN REVERSE ORDER
862 ; WE CAN'T USE THE MACRO SINCE THE P1 VECTORS WILL DISAPPEAR
863 ;       $DELTVA_S       (R2),8(R2)      ; DELETE FROM KERNEL STACK TO VECTORS
864         PUSHL   #0
865         PUSHAQ  8(R2)
866         PUSHAQ  (R2)
867         MOVAB   G^P1SYSVECTORS,R0
868         MNEGL   R0,R0
869         MOVAB   G^SYS$DELTVA,R1
870         CALLS   #3,^X80000000(R1)[R0]   ; DELETE KSTK TO VECTORS
871         BRW     DEL_PAGES               ; DELETE THE PAGES
872                                         ; ENTER & EXIT DELTVA AT IPL(ASTDEL)
873         .SAVE_PSECT
874         DECLARE_PSECT   EXEC$NONPAGED_CODE
875
876 DEL_PAGES:
877         LOCK    LOCKNAME=MMG            ; Lock MMG database
878         LOCK    LOCKNAME=SCHED          ; Lock SCHED database
879         MOVL    PCB$L_PHD(R4),R5        ; Get process header address
880         MOVZWL  PHD$W_PHVINDEX(R5),R10  ; Get balance slot index
881         ASSUME PHD$S_PRCPGFL EQ 4
882         PUSHL   PHD$B_PRCPGFL(R5)       ; Save page file mapping
883 ;
884 ;       Release all page table pages at this time
885 ;
886         BISW    #PHD$M_NO_WS_CHNG,-     ; Tell the swapper - hands off
887                 PHD$W_FLAGS(R5)
888         BICL3   #^X80000000,-
889                 PHD$L_P0BR(R5),R6       ; Start at base of P0 table
890         ASHL    #-7,R6,R6               ; Offset in SPT
```

```
  891          ADDL    G^MMG$GL_SPTBASE,R6
  892          SUBL3   G^SGN$GL_PHDPAGCT,-     ; Get count of page table pages
  893                  G^SWP$GL_BSLOTSZ,R7     ;  pages in the balance slot
  894          MOVL    PHD$L_FREPTECNT(R5),R9  ; Free PTEs
  895          ASHL    #-7,R9,R9               ; Number of SPT entries for free PTEs
  896          SUBL3   R9,R7,R9                ; Number of SPT entries not free
  897          MOVL    #-101,R8                ; Max to clean up with locks held
  898 10$:     EXTZV   #PTE$V_PFN,#PTE$S_PFN,-
  899                  (R6),R0                 ; Get the PFN
  900          BEQL    130$                    ; Empty
  901          INCL    R8                      ; Count one -  we are going to do work
  902          BEQL    200$                    ; Go release locks temporarily
  903          MOVL    (R6)+,R1                ; Get the PTE
  904          BLSS    150$                    ; Valid
  905          BBS     #PTE$V_TYP1,R1,50$      ; Backing store
  906          EXTZV   #PFN$V_LOC,#PFN$S_LOC,- ; Get the page location
  907                  @W^PFN$AB_STATE[R0],R2
  908          ASSUME  PFN$C_FREPAGLST EQ 0    ;
  909          ASSUME  PFN$C_MFYPAGLST EQ 1    ;
  910          ASSUME  PFN$C_BADPAGLST EQ 2    ;
  911          ASSUME  PFN$C_RELPEND   EQ 3    ;
  912          ASSUME  PFN$C_RDERR     EQ 4    ;
  913          ASSUME  PFN$C_WRTINPROG EQ 5    ;
  914          ASSUME  PFN$C_RDINPROG  EQ 6    ;
  915          ASSUME  PFN$C_ACTIVE    EQ 7    ;
  916          CASE    R2,<-                   ; Dispatch on page location
  917                  30$,-                   ; 0 => FREE PAGE LIST
  918                  30$,-                   ; 1 => MODIFIED PAGE LIST
  919                  30$,-                   ; 2 => BAD PAGE LIST, PAGE READ/WRITE ERR
  920                  400$,-                  ; 3 => RELEASE PENDING
  921                  400$,-                  ; 4 => PAGE READ ERROR
  922                  300$,-                  ; 5 => WRITE IN PROGRESS
  923                  400$,-                  ; 6 => READ IN PROGRESS
  924                  250$>                   ; 7 =>  ACTIVE
  925
  926 30$:     JSB     G^MMG$REMPFN            ; Remove it from the appropriate list
  927 40$:     JSB     G^MMG$DELCONPFN         ; Get rid of all traces of its contents
  928          CLRW    @W^PFN$AW_REFCNT[R0]    ; Ref count is now zero
  929          CLRL    R2                      ; Indicate free list
  930          JSB     G^MMG$INSPFNH           ; Put it on the list
  931          MOVL    -4(R6),R1               ; Get the PTE again
  932 50$:     EXTZV   #PTE$V_PGFLVBN,-        ; Get page file VBN
  933                  #PTE$S_PGFLVBN,R1,R0
  934          BEQL    90$                     ; None
  935          EXTZV   #PTE$V_PRCPGFLX,-       ; Get process specific page file index
  936                  #PTE$S_PRCPGFLX,R1,R3
  937          MOVZBL  (SP)[R3],R3             ; System wide page file index
  938          JSB     G^MMG$DALCPAGFIL        ; Release it
  939 90$:     CLRL    -4(R6)                  ; Clean up the PTE
  940 100$:    SOBGTR  R7,10$                  ; Go do the next one
  941          ADDL    #4,SP                   ; Clean up the stack
  942          BRW     DELETE_PERM
  943
  944 130$:    CLRL    (R6)+
  945          BRB     100$
  946
  947 150$:    BRB     250$
```

```
 948
 949 200$:   UNLOCK  LOCKNAME=MMG            ; Unlock MMG database
 950         UNLOCK  LOCKNAME=SCHED          ; Unlock SCHED database
 951         SETIPL  #IPL$_ASTDEL
 952         MOVL    #-101,R8                ; Reset  counter
 953         BRB     350$
 954 ;
 955 ;       Here we are  taking care of valid page table pages.
 956 ;       If they are not guaranteed to be totally free, we must be careful
 957 ;       not to free any with pages locked permanently in the working set
 958 ;       since those pages still exist.  The only ones we can be sure of
 959 ;       are ones with an outstanding modified page write.
 960 ;
 961 250$:   CMPW    @W^PFN$AW_REFCNT[R0],#1 ; Are there any extra references
 962         BNEQ    300$                    ; Assume modified page write
 963         CMPL    R9,R7                   ; Are we still in the free page tables
 964         BGEQ    100$                    ; No - leave valid pages alone
 965         BICB    #<<PTE$M_VALID!PTE$M_MODIFY>@-24>,-
 966                 -1(R6)                  ; Make it a transition PTE
 967         INCW    @PHV$GL_REFCBAS[R10]    ; Must up ref count before deleting
 968         BRW     40$
 969
 970
 971 ;
 972 ;       We need to wait for the modified page writer to complete
 973 ;
 974 300$:   SUBL    #4,R6                   ; Start again with the same PTE
 975         MOVL    #RSN$_MPWBUSY,R1        ; Resource
 976         JSB     G^MMG$RESRCWAIT         ; Get us on the right queue
 977         MOVPSL  -(SP)                   ; Use this PSL for wait
 978         JSB     G^MMG$SVPCTX            ; Go wait
 979 350$:   LOCK    LOCKNAME=MMG            ; Lock MMG database
 980         LOCK    LOCKNAME=SCHED          ; Lock SCHED database
 981         BRW     10$                     ; Try again
 982
 983
 984 400$:   BUG_CHECK ICPAGELOC,FATAL       ; INCONSISTENT PAGE LOCATION
 985
 986
 987 DELETE_PERM:
 988         .ENABLE LSB
 989
 990         SVPCTX                          ; SAVE PROCESS CONTEXT
 991                                         ; AND GO TO INTERRUPT STACK
 992         find_cpu_data   R3,ISTACK=YES
 993
 994 ;-------------------------------------------------------------
 995 ;
 996 ;       RESET PROCESS AFFINITY CHARACTERISTICS
 997 ;
 998 ;-------------------------------------------------------------
 999
1000         BBC     #CPB$V_EXPLICIT_AFFINITY,-
1001                 PCB$L_CAPABILITY(R4),100$; check for explicit affinity set
1002         DECW    CPU$W_HARDAFF(R3)       ; decrement hard affinity count
1003 100$:   CLRL    PCB$L_CAPABILITY(R4)    ; zap required capabilities,
1004
```

```
1005          MOVL    G^SCH$AR_NULLPCB,CPU$L_CURPCB(R3) ; SET NULL AS CURRENT PCB
1006          MOVZWL  @W^PHV$GL_PIXBAS[R10],R1 ; GET PROCESS INDEX
1007          MOVL    G^SCH$AR_NULLPCB,@W^SCH$GL_PCBVEC[R1] ; RELEASE PROCESS INDEX SLOT
1008          MOVL    PHD$L_WSLOCK(R5),R9     ; POINT TO FIRST NON PERM PAGE
1009          CLRL    (R5)[R9]                ; MARK WITH A ZERO
1010          MOVL    PHD$L_WSLIST(R5),R8     ; LONGWORD INDEX TO START OF WORKING SET LIS
1011          MOVAL   (R5)[R8],R8             ; FORM BYTE ADDRESS OF START OF WSLIST
1012 130$:    MOVL    (R8)+,R2                ; GET VA OF PERM PAGE
1013          BEQL    140$                    ; DONE, NONE LEFT
1014          BLBC    R2,130$                 ; IGNORE IF NOT VALID
1015          BLSS    130$                    ; OR IF HEADER PAGES (SYSTEM ADDRESS)
1016          BSBW    DELPAGE                 ; DELETE PAGE
1017          BRB     130$                    ; CONTINUE
1018
1019 140$:    REMQUE  @PCB$L_ASTQFL(R4),R0    ; REMOVE PENDING AST CONTROL BLOCKS
1020          BVS     150$                    ; NONE LEFT
1021          BBS     #ACB$V_NODELETE,-       ; BR. IF NODELETE BIT IS SET.  THIS
1022                  ACB$B_RMOD(R0),140$     ; SHOULD NOT HAPPEN BUT IF IT DOES MAY
1023                                          ; RESULT IN LOSING POOL (WHICH IS
1024                                          ; PREFERABLE OVER A DOUBLE DEALLOCATE).
1025          BSBB    DEANONPAGED             ; DEALLOCATE SPACE FOR ACB
1026          BRB     140$                    ; AND CONTINUE
1027
1028 DEANONPAGED:                             ;
1029          JMP     G^EXE$DEANONPAGED       ; JUMP TO REACH REAL DEANONPAGED
1030
1031 ; The CWPSSRV$ structures for any ongoing CWPS operations will be queued to the
1032 ; PCB.  Since these structures will be referenced by the fork thread which will
1033 ; resume when the response from the remote system arrives, it would be bad form
1034 ; to deallocate the CWPSSRV$ structures now.  When the response arrives, the
1035 ; fork thread will realize that the requestor process (that's us) is gone, and
1036 ; it will deallocate the structure.
1037 ;
1038 ; So, if we find any of these we will simply remove them from our PCB and hang
1039 ; them from the SWAPPER'S PCB.  The SWAPPER won't notice them, and that both gives
1040 ; us a way to locate any LOST CWPSSRV$ structures, and means that we don't have
1041 ; to special case the REMQUE in the deallocate routine (the CWPSSRV$ will always
1042 ; be part of a valid queue).
1043 ;
1044 ; The CWPS fork thread will acquire the SCHED lock to synchronize with this
1045 ; activity (we have it now).
1046
1047 150$:    REMQUE  @PCB$Q_CWPSSRV_QUEUE(R4),R0     ; REMOVE PENDING CWPSSRV$ BLOCKS
1048          BVS     160$                    ; NONE LEFT
1049          MOVL    G^SCH$AR_SWPPCB,R1      ; GET ADDRESS OF SWAPPER PCB
1050          INSQUE  (R0),@PCB$Q_CWPSSRV_QUEUE(R1)   ; HANG FROM THE SWAPPER
1051          BRB     150$                    ; AND TRY NEXT
1052
1053 160$:    MOVL    PCB$Q_PRIV+ARB$L_RIGHTSLIST+8(R4),R0 ; GET EXTENDED RIGHTS LIST
1054          BEQL    170$                    ; BRANCH IF NONE
1055          BSBB    DEANONPAGED             ; DEALLOCATE IT
1056 170$:    MOVL    PCB$L_JIB(R4),R0        ; GET ADDRESS OF JIB FOR RELEASE
1057          BEQL    180$                    ; BR IF NONE
1058          ADAWI   #-1,JIB$W_PRCCNT(R0)    ; ONE LESS PROCESS IN THIS TREE
1059          CMPL    PCB$L_PID(R4),JIB$L_MPID(R0) ; IS THIS THE MASTER PROCESS?
1060          BNEQ    180$                    ; BR IF NOT
1061          BSBB    DEANONPAGED             ; DEALLOCATE JIB
```

```
SYSDELPRC - DELETE PROCESS SYSTEM SERVICE 10-MAY-1989 16:40:14 VAX MACRO V5.0-8 Page 21
X-33 DELETE - PERFORM DELETE ACTIONS IN CONTE 20-DEC-1988 12:59:50 [SYS.SRC]SYSDELPRC.MAR;1

 1062 180$:    MOVZWL  PCB$L_OWNER(R4),R0       ; GET PIX FROM OWNER UIC
 1063          BEQL    190$                     ; BR IF NOT A SUB-PROCESS
 1064          MOVL    @W^SCH$GL_PCBVEC[R0],R0  ; GET ADDRESS OF OWNER PCB
 1065          DECW    PCB$W_PRCCNT(R0)         ; DECREMENT COUNT OF SUB-PROCESSES
 1066          MOVL    #RSN$_ASTWAIT,R0         ; NOTIFY PARENT PROCESS THAT ITS
 1067          JSB     G^SCH$RAVAIL             ;  SUBPROCESS COUNT HAS BEEN DECREMENTED
 1068 190$:    MOVL    R4,R0                    ; SET ADDRESS OF PCB FOR RELEASE
 1069          BSBB    DEANONPAGED              ; AND DELETE IT
 1070          DECW    G^SWP$GW_BALCNT          ; DECREASE COUNT IN BALANCE SET
 1071          JSB     G^SCH$SWPWAKE            ; WAKE SWAPPER
 1072          MCOMW   #0,@W^PHV$GL_PIXBAS[R10] ; INDICATE PROCESS HEADER TO BE DELETED
 1073          INCW    G^SCH$GW_DELPHDCT        ; FLAG SWAPPER'S ATTENTION
 1074          DECW    G^SCH$GW_PROCCNT         ; ONE LESS PROCESS NOW
 1075          UNLOCK  LOCKNAME=MMG             ; UNLOCK MMG DATABASE
 1076          JMP     G^SCH$SCHED              ; AND SCHEDULE FOR ANOTHER PROCESS
 1077
 1078          .DISABLE LSB
```

```
1080          .SBTTL  DELPAGE - DELETE PAGE
1081 ;++
1082 ; FUNCTIONAL DESCRIPTION:
1083 ;       DELPAGE DELETES THE CONTENTS OF THE SPECIFIED PAGE AND PLACES
1084 ;        THE PFN ON THE FREE PAGE LIST.
1085 ;
1086 ; CALLING SEQUENCE:
1087 ;       BSBx    DELPAGE
1088 ;
1089 ; INPUT PARAMETERS:
1090 ;       R2 - VIRTUAL ADDRESS OF PAGE TO RELEASE
1091 ;       R4 - PCB ADDRESS OF PROCESS OWNING PAGE
1092 ;       R5 - PROCESS HEADER ADDRESS OF PROCESS OWNING PAGE
1093 ;
1094 ; OUTPUT PARAMETERS:
1095 ;       NONE
1096 ;
1097 ; ENVIRONMENT:
1098 ;       IPL = SYNCH, with MMG spinlock acquired.
1099 ;
1100 ;--
1101
1102 DELPAGE:                                ; DELETE PAGE
1103          JSB     G^MMG$SVAPTECHK        ; GET SVA OF PTE FOR PAGE
1104          BBCC    #PTE$V_VALID,(R3),50$  ; EXIT IF NOT VALID
1105          BBCC    #PTE$V_MODIFY,(R3),10$ ; CLEAR MODIFY INDICATOR
1106 10$:     EXTZV   #0,#PTE$S_PFN,(R3),R0  ; GET PFN FROM PTE
1107          DECREF  ,,,L^,,JSB             ; DROP REFERENCE COUNT ON PAGE
1108          BGTR    50$                    ; BR IF CANT RELEASE YET
1109          BICB    #PFN$M_MODIFY,@W^PFN$AB_STATE[R0] ; MAKE SURE NO MODIFY BIT
1110 20$:     JSB     G^MMG$DELCONPFN        ; DELETE CONTENT OF PFN
1111          ASSUME  PFN$C_FREPAGLST EQ 0   ;
1112          CLRL    R2                     ; INDICATE FREE LIST
1113          JSB     G^MMG$INSPFNH          ; INSERT AT HEAD OF FREE LIST
1114 50$:     RSB                            ;
1115
1116          .RESTORE_PSECT
```

```
1118          .SBTTL  TERMMBX - SEND MESSAGE TO TERMINATION MAILBOX
1119
1120 ;----------------------------------------------------------------------
1121 ;
1122 ;       BUILD PROCESS TERMINATION MESSAGE AND SEND TO TERMINATION
1123 ;       MAILBOX UNIT.
1124 ;
1125 ;----------------------------------------------------------------------
1126
1127 TERMMSG:                                ; SEND TERMINATION MESSAGES
1128          MOVL    SP,R10                 ; SAVE STACK STATE
1129          PUSHL   PCB$L_EOWNER(R4)       ; SEND EXTENDED PID OF SUBPROCESS OWNER (0 =
1130          MOVL    G^CTL$GL_PHD,R5        ; GET PROCESS HEADER ADDRESS
1131          MOVQ    G^CTL$GQ_LOGIN,-(SP)   ; LOGIN TIME
1132          ASSUME  ACC$L_OWNER EQ ACC$Q_LOGIN+8
1133          PUSHL   G^CTL$GL_VOLUMES       ; COUNT OF MOUNTED VOLUMES
1134          ASSUME  ACC$Q_LOGIN EQ ACC$L_VOLUMES+4
1135          PUSHL   PHD$L_DIOCNT(R5)       ; DIRECT I/O COUNT
1136          ASSUME  ACC$L_VOLUMES EQ ACC$L_DIOCNT+4
1137          PUSHL   PHD$L_BIOCNT(R5)       ; BUFFERED I/O COUNT
1138          ASSUME  ACC$L_DIOCNT EQ ACC$L_BIOCNT+4
1139          PUSHL   G^CTL$GL_WSPEAK        ; PEAK WORKING SET SIZE
1140          ASSUME  ACC$L_BIOCNT EQ ACC$L_WSPEAK+4
1141          PUSHL   #0                     ; PEAK PAGING FILE USAGE
1142          ASSUME  ACC$L_WSPEAK EQ ACC$L_PGFLPEAK+4
1143          PUSHL   PHD$L_PAGEFLTS(R5)     ; TOTAL PROCESS PAGE FAULTS
1144          ASSUME  ACC$L_PGFLPEAK EQ ACC$L_PAGEFLTS+4
1145          PUSHL   PHD$L_CPUTIM(R5)       ; SAVE ACCUMULATED CPU TIME FOR PROCESS
1146          ASSUME  ACC$L_PAGEFLTS EQ ACC$L_CPUTIM+4
1147          MOVAB   G^CTL$T_USERNAME+12,R0 ; GET BASE ADDRESS
1148          MOVQ    -(R0),-(SP)            ; LAST HALF OF USER NAME
1149          PUSHL   -(R0)                  ; FIRST LONGWORD OF USER NAME
1150          MOVAB   W^<CTL$T_ACCOUNT-CTL$T_USERNAME>(R0),R0 ; POINT PAST ACCOUNT
1151          ASSUME  ACC$L_CPUTIM EQ ACC$T_USERNAME+12
1152          MOVQ    (R0),-(SP)             ; SET ACCOUNT INTO RECORD
1153          ASSUME  ACC$T_USERNAME EQ ACC$T_ACCOUNT+8
1154          MOVQ    G^EXE$GQ_SYSTIME,-(SP) ; LOGOUT TIME
1155          ASSUME  ACC$T_ACCOUNT EQ ACC$Q_TERMTIME+8
1156          PUSHL   #0                     ; JOBID
1157          ASSUME  ACC$Q_TERMTIME EQ ACC$L_JOBID+4
1158          PUSHL   PCB$L_EPID(R4)         ; EXTENDED PROCESS ID
1159          ASSUME  ACC$L_JOBID EQ ACC$L_PID+4
1160          PUSHL   G^CTL$GL_FINALSTS      ; FINAL STATUS OF PROCESS
1161          ASSUME  ACC$L_PID EQ ACC$L_FINALSTS+4
1162          PUSHL   #MSG$_DELPROC          ; MESSAGE TYPE
1163          ASSUME  ACC$L_FINALSTS EQ ACC$W_MSGTYP+4
1164          MOVL    SP,R7                  ; MESSAGE ADDRESS
1165          MOVZWL  PCB$W_TMBU(R4),R0      ; TERMINATION MAILBOX UNIT
1166          MOVQ    #^A/MB00000:/,-(SP)    ; SKELETON FOR DEVICE STRING
1167          CLRL    R1                     ; INIT FOR EDIV
1168          MOVAB   7(SP),R3               ; POINTER TO LSD OF DEVICE NAME
1169 10$:     EDIV    #10,R0,R0,R2           ; CONVERT A DECIMAL DIGIT
1170          ADDB    R2,-(R3)               ; SET DIGIT INTO NAME STRING IN ASCII
1171          TSTL    R0                     ; CHECK FOR END
1172          BNEQ    10$                    ; NOT YET, CONVERT ANOTHER DIGIT
1173          PUSHL   SP                     ; BUILD STRING DESCRIPTOR FOR
1174          PUSHL   #8                     ; MAILBOX NAME
```

```
1175          MOVL    SP,R8                   ; AND SAVE POINTER TO NAME DESCRIPTOR
1176          MOVAL   -(SP),R9                ; RESERVE SPACE FOR CHANNEL
1177 ;
1178 ; CHECK FOR PRESENCE OF CHANNEL TABLE, IF NON-EXISTENT (I.E. PROCESS DELETED
1179 ; BEFORE THE CREATION OF IT) USE A PAGE OF SUPERVISOR STACK FOR IT.
1180 ;
1181          TSTW    G^CTL$GW_NMIOCH         ; IS THERE A VALID CHANNEL TABLE?
1182          BNEQ    20$                     ; YES, SKIP TO ASSIGN
1183          PUSHL   R4                      ; SAVE PCB ADDRESS
1184          MOVL    G^CTL$AL_STACK+<4*PSL$C_SUPER>,R3       ; PICK UP SUPER SP
1185          MOVAB   -CCB$C_LENGTH(R3),-
1186                  G^CTL$GL_CCBBASE        ; MAKE IT TABLE BASE
1187          MOVC5   #0,(SP),#0,#512,-512(R3); CLEAR IT OUT
1188          POPL    R4                      ; RESTORE PCB ADDRESS
1189          MOVZWL  #<512/CCB$C_LENGTH>-1,- ; SET NUMBER OF CHANNELS
1190                  G^CTL$GW_NMIOCH
1191 ;
1192 ; ASSIGN CHANNEL TO MAILBOX AND WRITE THE MESSAGE
1193 ;
1194 20$:
1195          $ASSIGN_S       DEVNAM=(R8),-   ; ATTEMPT TO ASSIGN CHANNEL TO
1196                          CHAN=(R9)       ; MAILBOX
1197          BLBC    R0,30$                  ; UNABLE TO ASSIGN, IGNORE
1198          $QIO_S          CHAN=(R9),-     ; SEND TERMINATION MESSAGE
1199                          FUNC=#<IO$_WRITEVBLK!IO$M_NOW>,-
1200                          P1=(R7),-       ; MESSAGE ADDRESS
1201                          P2=#ACC$C_TERMLEN; MESSAGE SIZE
1202 30$:
1203          $DASSGN_S       CHAN=(R9)       ; DEASSIGN CHANNEL
1204          MOVL    R10,SP                  ; RESTORE STACK STATE
1205          RSB
1206
```

```
1208          .SUBTITLE       Return Unused CPU Time Limit to Parent
1209
1210 ;+
1211 ; Functional Description:
1212 ;
1213 ;       RETQUOTA is entered via a special kernel AST directed to the
1214 ;       creator of a subprocess.  The AST control block contains the
1215 ;       unused CPU time limit to be returned to the creator.
1216 ;
1217 ;       The CPU limit of the parent has any unused CPU time from the
1218 ;       subprocess added to it.
1219 ;
1220 ; Input Parameters:
1221 ;
1222 ;       R4      PCB Address of Current (Parent) Process
1223 ;       R5      Pointer to AST Control Block
1224 ;
1225 ;       ACB_L_CPULIM(R5)        Original CPU Limit of Subprocess
1226 ;       ACB_L_CPUTIM(R5)        CPU Time Used by Subprocess
1227 ;
1228 ; Output Parameter:
1229 ;
1230 ;       PHD$L_CPULIM(R1)        The CPU limit of the parent is adjusted
1231 ;                               according to the following rules.
1232 ;
1233 ;       If the subprocess has unused CPU time limit (a credit) ,
1234 ;       that credit is added to the parent's CPU limit.
1235 ;
1236 ;       If the subprocess exceeded its CPU time allotment, an attempt
1237 ;       is made to pass the deficit to the parent.
1238 ;
1239 ;               If the deficit is smaller than the parent's CPU limit, the
1240 ;               deficit is simply subtracted from CPU limit.
1241 ;
1242 ;               If the deficit is larger than the parent's CPU limit, the
1243 ;               parent is forced into the CPU time expiration path by
1244 ;               loading its CPUTIM into its CPULIM field, indicating
1245 ;               that its CPU time limit has expired. (Note that loading CPULIM
1246 ;               into CPUTIM has the same effect but causes double accounting
1247 ;               for the CPU time already charged to the subprocess.)
1248 ;
1249 ; Side Effects:
1250 ;
1251 ;       The AST control block is released to nonpaged pool.
1252 ;
1253 ; Environment:
1254 ;
1255 ;       IPL$_ASTDEL while carrying pointer to AST control block.
1256 ;-
1257
1258 RETQUOTA:
1259          MOVL    G^CTL$GL_PHD,R1         ; Get safe pointer to process header
1260          TSTL    PHD$L_CPULIM(R1)        ; Check for no quota
1261          BEQL    20$                     ; Skip return if no quota
1262          SUBL3   ACB_L_CPUTIM(R5),ACB_L_CPULIM(R5),R0 ; Get difference
1263          BGEQU   10$                     ; Excess or deficit (Branch if excess)
1264          MNEGL   R0,-(SP)                ; Handle deficit very carefully
```

```
 1265          CMPL    (SP)+,PHD$L_CPULIM(R1)  ; Can creator accommodate deficit?
 1266          BLSSU   10$                     ; Branch if yes
 1267          MOVL    PHD$L_CPUTIM(R1),PHD$L_CPULIM(R1) ; Otherwise, force CPU time
 1268                                          ;  expiration by setting limit to time
 1269          BRB     20$                     ; Deallocate AST control block
 1270
 1271 10$:     ADDL2   R0,PHD$L_CPULIM(R1)     ; Give back excess
 1272 20$:     MOVL    R5,R0                   ; Get address for release
 1273          JMP     G^EXE$DEANONPAGED       ; Return via EXE$DEANONPAGED
 1274
 1275          .END
```

16 SWAPPER.LIS

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 0
Table of contents
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 1
X-35U3 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (1)

    1          .TITLE  SWAPPER  WORKING SET SWAPPER
    2          .IDENT  'X-35U3'
    3 ;
    4 ;*****************************************************************************
    5 ;*                                                                           *
    6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
    7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
    8 ;*  ALL RIGHTS RESERVED.                                                     *
    9 ;*                                                                           *
   10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
   11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
   12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
   13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
   14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
   15 ;*  TRANSFERRED.                                                             *
   16 ;*                                                                           *
   17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
   18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
   19 ;*  CORPORATION.                                                             *
   20 ;*                                                                           *
   21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
   22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
   23 ;*                                                                           *
   24 ;*                                                                           *
   25 ;*****************************************************************************
   26
   27 ;++
   28 ; FACILITY: EXECUTIVE, SWAPPER
   29 ;
   30 ; ABSTRACT: THE SWAPPER SCHEDULES AND EXECUTES SWAPPING OF PROCESS
   31 ;           WORKING SETS BETWEEN SWAP STORAGE AND MAIN MEMORY.
   32 ;
   33 ; ENVIRONMENT:
   34 ;       MODE = KERNEL ,  RESIDENT
   35 ;
   36 ; AUTHOR:  R. HUSTVEDT   CREATION DATE: 30-NOV-76
   37 ;
   38 ; MODIFIED BY:
   39 ;
   40 ;       X-35U3  SSA0010         Stan Amway              20-Mar-1989
   41 ;               Properly dispose of page file space in DELPHD.
   42 ;               The code to do so was (correctly) removed in X-25.
   43 ;               However, it should have been reinstated (probably
   44 ;               around X-27) to track a subsequent change in SYSDELPRC.
   45 ;
   46 ;       X-35U2  SSA0009         Stan Amway              10-Aug-1988
   47 ;               Avoid flushing MPL to low limit when possible.
   48 ;
   49 ;       X-35U1  SSA0008         Stan Amway              15-Jul-1988
   50 ;               Use alternate PQL list if booting a stand-alone application.
   51 ;
   52 ;       X-35    SSA0007         Stan Amway              12-Apr-1988
   53 ;               Selectively flush modified page list.
   54 ;
   55 ;       X-34    SSA0006         Stan Amway              28-Oct-1987
   56 ;               Specify PQL list when creating SYSINIT.
   57 ;
```

```
;       X-33    SF00033         Stephen Fiorelli        26-Aug-1987
;               Final large working set support.
;
;       X-32    SSA0005         Stan Amway               4-Aug-1987
;        -31    Swap file allocation changes.
;
;       X-30    SJF             Stu Farnham             30-Jul-1987
;               INVALID becomes INVALIDATE_TB. Make TBIS track SRM
;               rev H.
;
;       X-29    SF00028         Stephen Fiorelli        27-Jul-1987
;               Initial large working set support.
;
;       X-28    SSA0004         Stan Amway              22-Apr-1987
;               Fix BAK setup in PHD inswap.
;               (Bug incorrectly fixed in X-24.)
;
;       X-27    WMC0027         Wayne Cardoza           30-Mar-1987
;               Scan more of balance slot for now.  Calculation was
;               incorrect.  It will be fixed in the future.
;
;       X-26    SF04007         Stephen Fiorelli        09-Mar-1987
;               Use vms standard name for system_primitives_data.
;
;       X-25    WMC0025         Wayne Cardoza           05-Mar-1987
;               Don't scan most of balance slot in DELPHD.
;
;       X-24    WMC0024         Wayne Cardoza           05-Mar-1987
;               Fix BAK setup in PHD  inswap.
;
;       X-23    SSA0003         Stan Amway              27-Feb-1987
;               Add bugcheck in DELPHD processing to catch bad
;               process to system page file mapping.
;
;       X-22    WCT0032         Ward C. Travis          19-Feb-1987
;               Update remaining old lookaside  listhead references
;               to reflect that they are now interlocked queues.
;
;       X-21    WMC0021         Wayne Cardoza           16-Feb-1987
;               Free page cleanup now done in DELPRC.
;
;       X-20    SF04006         Stephen Fiorelli        05-Feb-1987
;               MMG$AL_SYSPCB becomes MMG$AR_SYSPCB.
;
;       X-19    MSH0293         Michael S. Harvey       28-Jan-1987
;               Correct mechanism that triggers powerfail AST declaration
;               so that it works under SMP.
;
;       X-18    SSA0002         Stan Amway               4-Oct-1986
;               Add support for multiple pagefiles per process.
;
;       X-17    RNG0017         Rod Gamache             29-Oct-1986
;               Fix branch error.
;
;       X-16    RNG0016         Rod Gamache              8-Oct-1986
;               Acquire SCHED lock after call to MMG$UNLOCK.
;
```

```
;       X-15    SSA0001         Stan Amway              11-Sep-1986
;               Allow swap I/O greater than 127 pages for devices
;               that support it.
;
;               Account for modified pages in transit when checking
;               free page count. This finesse is important now that
;               modified page writing is multi-threaded.
;
;               Don't start modified page writing to balance free page
;               deficit unless the number of modified pages exceeds
;               the SYSGEN threshold value, MPW_THRESH.
;
;               Avoid saving R0 on most calls to SMPLOCK and SMPUNLOCK.
;
;               Move reset of swap failure counter from SWAPSCHED to
;               INSWAP. This insures that the reset is only done when
;               an inswap is actually successful.
;
;       X-14    SF04005         Stephen Fiorelli        11-Aug-1986
;               Eliminate use of swp$gl_shelio.  Use swp$c_shellsiz
;               instead.
;
;       X-13    WMC0005         Wayne Cardoza           03-Sep-1986
;               Use LDR$INIT_ALL.
;
;       X-12    WMC0004         Wayne Cardoza           25-Jul-1986
;               Temporary fix to call initialization in correct order.
;
;       X-10    SJF             Stu Farnham             8-Jul-1986
;               Fix merge error
;
;       X-6     WMC0003         Wayne Cardoza           12-May-1986
;               Make process context initialization code paged.
;
;       X-4E5   HH0182          Hai Huang               24-Apr-1986
;               Correctly set up PQL$GL_SYSPQLLEN.
;
;       X-5     SF04004         Stephen Fiorelli        02-Apr-1986
;               Resolve conflicts from merge of exec_reorg_V4.4 into
;               mainline.
;
;       X-4     ACG0505         Andrew C. Goldstein,    6-Nov-1985  18:24
;               Deny control access to group and world in log name protection
;
;       X-3D5   HH0158          Hai Huang               13-Mar-1986
;               Add support for shadowed system disk.
;
;       X-3D4   SF04003         Stephen Fiorelli        03-Feb-1986
;               Change EXE$K_ references to BASE$EXE$K_.
;
;       X-3D3   SF04002         Stephen Fiorelli        30-Jan-1986
;               Make a reference to SGN$GL_PHDPAGCT general addressing.
;
;       X-3D2   TCM0008         Trudy C. Matthews       30-Jan-1986
;               Change SYS$K_CREPRC and SYS$K_CRELNM symbols to EXE$K_CREPRC
;               and EXE$K_CRELNM.
;
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 4
X-3503 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (1)

  172 ;       X-3D1   SF04001         Stephen Fiorelli        10-Dec-1985
  173 ;               Resolve conflicts from initial merge of exec reorg
  174 ;               thread and mainline (4.4 BL7).
  175 ;
  176 ;       X-1C11  TCM0007         Trudy C. Matthews       19-Nov-1985
  177 ;               Make PQL$GL_SYSPQLLEN a longword that contains the length
  178 ;               of PQL$AB_SYSPQL, and lives in the base image.
  179 ;               PQL$C_SYSPQLLEN will continue to be available as a local
  180 ;               constant to the SWAPPER.
  181 ;
  182 ;       X-1C10  TCM0006         Trudy C. Matthews       18-Nov-1985
  183 ;               Change names of pointers to structures that live in the
  184 ;               SWAPPER, so other modules know that these structures are now
  185 ;               accessed indirectly through a pointer:
  186 ;
  187 ;               LNM$SYSTEM_DIRECTORY's pointer: LNM$AL_SYSTEM_DIRECTORY
  188 ;               LNM_SYSTEM_DIR_LNMTH's pointer: LNM_AL_SYSTEM_DIR_LNMTH
  189 ;               PQL$AB_SYSPQL's pointer:        PQL$AR_SYSPQL
  190 ;               PQL$C_SYSPQLLEN's pointer:      PQL$AR_SYSPQLLEN
  191 ;
  192 ;       X-1C9   TCM0005         Trudy C. Matthews       11-Nov-1985
  193 ;               Remove G^ from LNM$SYSTEM_DIRECTORY ASCII string.
  194 ;               Also, fix up LNM$AL_DIRTBL's pointer to LNM$SYSTEM_DIRECTORY.
  195 ;               (Make the pointer point to the directory table, not its
  196 ;               pointer in SYSTEM_DATA_CELLS).
  197 ;
  198 ;       X-1C8   TCM0004         Trudy C. Matthews       5-Nov-1985
  199 ;               Add ALIGNMENT=PAGE parameter to EXEC$NONPAGED_DATA psect.
  200 ;
  201 ;       X-1C6   TCM0003         Trudy C. Matthews       14-Oct-1985
  202 ;               Call all loaded system image's initialization routines again
  203 ;               from the SWAPPER.  This is so any .ADDRESS fixups in pagable
  204 ;               code can be done at this time.
  205 ;
  206 ;       X-1C5   TCM0002         Trudy C. Matthews       3-Oct-1985
  207 ;               Changed reference to PFN$AL_TAIL to G^.
  208 ;
  209 ;       X-1C4   TCM0001         Trudy C. Matthews       23-Aug-1985
  210 ;               Move all .ADDRESS and .ASCID directives to read/write
  211 ;               psects.
  212 ;
  213 ;       X-3     WMC0002         Wayne Cardoza           02-Aug-1985
  214 ;               Get empty page count correct if process header being expanded.
  215 ;
  216 ;       V04-002 WMC0001         Wayne Cardoza           12-Mar-1985
  217 ;               Scan page list backwards to reduce average time.
  218 ;
  219 ;       V04-001 WMC0001         Wayne Cardoza           28-Feb-1985
  220 ;               Don't continue scanning page lists after PHD ref count
  221 ;               goes to zero.
  222 ;
  223 ;--
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 5
X-35U3 DECLARATIONS 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (2)

  226          .SBTTL  DECLARATIONS
  227 ;
  228 ; INCLUDE FILES:
  229 ;
  230
  231          $ACBDEF                         ; DEFINE AST CONTROL BLOCK OFFSETS
  232          $BOOSTATEDEF                    ; DEFINE BOOTSTRAP STATE FLAGS
  233          $CPUDEF                         ; DEFINE PER-CPU DATA BLOCK OFFSETS
  234          $CREPRCDEF                      ; Define argument list for $CREPRC
  235          $DYNDEF                         ; DEFINE STRUCTURE TYPE CODES
  236          $IPLDEF                         ; DEFINE INTERRUPT PRIORITY LEVELS
  237          $IRPDEF                         ; DEFINE I/O REQUEST PACKET OFFSETS
  238          $LDRIMGDEF                      ; DEFINE LOADED SYSTEM IMAGE STRUCTURE
  239          $LNMDEF                         ; DEFINE LOGICAL NAME OFFSETS
  240          $LNMSTRDEF                      ; DEFINE lOGICAL NAME STRUCTURE OFFSETS
  241          $MPWDEF                         ; Define MPW structures and constants
  242          $OPDEF                          ; DEFINE OPCODE EQUIVALENT VALUES
  243          $ORBDEF                         ; DEFINE OBJECT RIGHTS BLOCK OFFSETS
  244          $PCBDEF                         ; DEFINE PCB OFFSETS
  245          $PFLDEF                         ; DEFINE SWAP FILE TABLE OFFSETS
  246          $PFLMAPDEF                      ; Define page/swap file map offsets
  247          $PFNDEF                         ; DEFINE PFN VALUES
  248          $PHDDEF                         ; DEFINE PHD OFFSETS
  249          $PQLDEF                         ; DEFINE QUOTA SYMBOLS
  250          $PRDEF                          ; DEFINE PROCESSOR REGISTERS
  251          $PRCDEF                         ; CREATE PROCESS FLAGS
  252          $PSLDEF                         ; DEFINE PSL VALUES
  253          $PTEDEF                         ; DEFINE PAGE TABLE ENTRY
  254          $RPBDEF                         ; DEFINE REBOOT PARAMETER BLOCK OFFSETS
  255          $SYSVECTORDEF                   ; DEFINE SYSTEM SERVICE OFFSETS
  256          $$SYSTEM_PRIM_DATADEF           ; DEFINE SYSTEM_PRIMITIVES PRIVATE DATA
  257          $UCBDEF                         ; DEFINE UCB OFFSETS
  258          $VADEF                          ; DEFINE VIRTUAL ADDRESS FIELDS
  259          $WSLDEF                         ; DEFINE WORKING SET LIST BITS
  260
  261 ;
  262 ; ASSUMPTIONS ABOUT THE STRUCTURE OF LOGICAL NAME AND OBJECT RIGHTS BLOCKS:
  263 ;
  264
  265          ASSUME  LNMB$L_FLINK,       EQ,  0
  266          ASSUME  LNMB$L_FLINK+4,     EQ,  LNMB$L_BLINK
  267          ASSUME  LNMB$L_BLINK+4,     EQ,  LNMB$W_SIZE
  268          ASSUME  LNMB$W_SIZE+2,      EQ,  LNMB$B_TYPE
  269          ASSUME  LNMB$B_TYPE+1,      EQ,  LNMB$B_ACMODE
  270          ASSUME  LNMB$B_ACMODE+1,    EQ,  LNMB$L_TABLE
  271          ASSUME  LNMB$L_TABLE+4,     EQ,  LNMB$B_FLAGS
  272          ASSUME  LNMB$B_FLAGS+1,     EQ,  LNMB$T_NAME
  273
  274          ASSUME  LNMX$B_FLAGS,       EQ,  0
  275          ASSUME  LNMX$B_FLAGS+1,     EQ,  LNMX$B_INDEX
  276          ASSUME  LNMX$B_INDEX+1,     EQ,  LNMX$W_HASH
  277          ASSUME  LNMX$W_HASH+2,      EQ,  LNMX$T_XLATION
  278
  279          ASSUME  LNMTH$B_FLAGS,      EQ,  0
  280          ASSUME  LNMTH$B_FLAGS+1,    EQ,  LNMTH$L_HASH
  281          ASSUME  LNMTH$L_HASH+4,     EQ,  LNMTH$L_ORB
  282          ASSUME  LNMTH$L_ORB+4,      EQ,  LNMTH$L_NAME
```

```
283          ASSUME  LNMTH$L_NAME+4,     EQ,  LNMTH$L_PARENT
284          ASSUME  LNMTH$L_PARENT+4,   EQ,  LNMTH$L_CHILD
285          ASSUME  LNMTH$L_CHILD+4,    EQ,  LNMTH$L_SIBLING
286          ASSUME  LNMTH$L_SIBLING+4,  EQ,  LNMTH$L_QTABLE
287          ASSUME  LNMTH$L_QTABLE+4,   EQ,  LNMTH$L_BYTESLM
288          ASSUME  LNMTH$L_BYTESLM+4,  EQ,  LNMTH$L_BYTES
289
290          ASSUME  ORB$L_OWNER,        EQ,  0
291          ASSUME  ORB$L_OWNER+4,      EQ,  ORB$L_ACL_MUTEX
292          ASSUME  ORB$L_ACL_MUTEX+4,  EQ,  ORB$W_SIZE
293          ASSUME  ORB$W_SIZE+2,       EQ,  ORB$B_TYPE
294          ASSUME  ORB$B_TYPE+1,       EQ,  ORB$B_FLAGS
295          ASSUME  ORB$B_FLAGS+3,      EQ,  ORB$W_REFCOUNT
296          ASSUME  ORB$W_REFCOUNT+2,   EQ,  ORB$Q_MODE_PROT
297          ASSUME  ORB$Q_MODE_PROT+8,  EQ,  ORB$L_SYS_PROT
298          ASSUME  ORB$L_SYS_PROT+4,   EQ,  ORB$L_OWN_PROT
299          ASSUME  ORB$L_OWN_PROT+4,   EQ,  ORB$L_GRP_PROT
300          ASSUME  ORB$L_GRP_PROT+4,   EQ,  ORB$L_WOR_PROT
301          ASSUME  ORB$L_WOR_PROT+4,   EQ,  ORB$L_ACL_COUNT
302          ASSUME  ORB$L_ACL_COUNT+4,  EQ,  ORB$L_ACL_DESC
303          ASSUME  ORB$L_ACL_DESC+4,   EQ,  ORB$R_MIN_CLASS
304          ASSUME  ORB$R_MIN_CLASS+ORB$S_MIN_CLASS,-
305                                      EQ,  ORB$R_MAX_CLASS
306          ASSUME  ORB$R_MAX_CLASS+ORB$S_MAX_CLASS,-
307                                      EQ,  ORB$K_LENGTH
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 7
X-35U3 DECLARATIONS 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (4)

  309
  310 ;
  311 ; OWN STORAGE:
  312 ;
  313
  314         DECLARE_PSECT   EXEC$NONPAGED_DATA,ALIGNMENT=PAGE
  315                                         ; SWAPPER/SCHEDULER WRITABLE DATA
  316 IOROUTINE:                             ; ADDRESS OF PROPER BUILD PACKET ROUTINE
  317         .LONG   0                       ;
  318 IOEA:   .LONG   0                       ; I/O END ACTION RETURN
  319 TPGCNT: .LONG   0                       ; Remaining page count (entire xfer)
  320 PFLMAP: .LONG   0                       ; Address of current mapping pointer
  321 RWSSWP: .LONG   0                       ; REMAINING WS SWP ADDRESS
  322 RSVAPTE:.LONG   0                       ; REMAINING SVA OF PTE
  323 RPGCNT: .LONG   0                       ; REMAINING PAGE COUNT (current pointer)
  324 OSWPPGS:.LONG   0                       ; OUTSWAP PAGE COUNT
  325 OSWPPCB:.LONG   0                       ; PCB ADDRESS OF OUTSWAP PROCESS
  326         UNIVERSAL_SYMBOL -              ; DEFINE SWP$GL_MAP AS UNIVERSAL SO
  327 SWP$GL_MAP                             ; IT MAY BE REFERENCED OUTSIDE THIS
  328         .LONG   0                       ; IMAGE
  329
  330
  331 ;
  332 ; LNM$SYSTEM_DIRECTORY - THE SYSTEM DIRECTORY LOGICAL NAME TABLE.
  333 ;
  334
  335         UNIVERSAL_SYMBOL        LNM$AR_SYSTEM_DIRECTORY
  336                                         ; This is the name of the pointer
  337                                         ; used by external modules
  338 LNM$SYSTEM_DIRECTORY:                   ; This is the name used by SWAPPER
  339         .LONG   0                       ; FORWARD LINK
  340         .LONG   0                       ; BACK LINK
  341         .WORD   LNM_SYS_DIR_SIZ         ; SIZE OF STRUCTURE
  342         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
  343         .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
  344         .ADDRESS LNM_SYSTEM_DIR_LNMTH   ; DIRECTORY TABLE HEADER ADDRESS
  345         .BYTE   LNMB$M_NO_ALIAS!-       ; DIRECTORY TABLES CAN NOT BE ALIASED
  346                 LNMB$M_TABLE!-          ; DIRECTORIES ARE TABLES
  347                 LNMB$M_NODELETE         ; DIRECTORIES CAN NOT BE DELETED
  348         .ASCIC  LNM$SYSTEM_DIRECTORY  ; NAME OF DIRECTORY TABLE

  349
  350         .BYTE   LNMX$M_TERMINAL         ; FLAGS BYTE. NO MORE TRANSLATIONS
  351         .BYTE   LNMX$C_TABLE            ; TRANSLATION INDEX ( SPECIAL TABLE )
  352         .WORD   0                       ; TRANSLATION HASH CODE
  353         .BYTE   LNMTH$K_LENGTH          ; SIZE OF TABLE HEADER BLOCK
  354
  355         UNIVERSAL_SYMBOL        LNM_AR_SYSTEM_DIR_LNMTH
  356                                         ; This is the name of the pointer
  357                                         ; used by other modules
  358 LNM_SYSTEM_DIR_LNMTH:                   ; This is the name used by SWAPPER
  359
  360         .BYTE   LNMTH$M_SHAREABLE!-     ; DIRECTORY IS A SHAREABLE TABLE
  361                 LNMTH$M_DIRECTORY       ; TABLE IS A DIRECTORY TABLE
  362         .LONG   0                       ; ADDRESS OF HASH TABLE
  363         .ADDRESS LNM_SYSTEM_DIR_ORB     ; ADDRESS OF OBJECT RIGHTS BLOCK
```

```
364         .ADDRESS LNM$SYSTEM_DIRECTORY   ; ADDRESS OF CONTAINING LNMB BLOCK
365         .LONG   0                       ; ADDRESS OF PARENT TABLE
366         .LONG   0                       ; ADDRESS OF CHILD TABLE
367         .LONG   0                       ; ADDRESS OF SIBLING TABLE
368         .ADDRESS LNM_SYSTEM_DIR_LNMTH   ; ADDRESS OF TABLE HOLDING QUOTA
369         .LONG   ^X7FFFFFFF              ; INITIAL QUOTA   ( POSITIVE INFINITY )
370         .LONG   ^X7FFFFFFF              ; REMAINING QUOTA ( POSITIVE INFINITY )
371
372         .BYTE   LNMX$M_XEND             ; FLAGS BYTE. NO MORE TRANSLATIONS
373
374         .ALIGN  QUAD
375 LNM_SYSTEM_DIR_ORB:
376         .LONG   ^X00010004              ; SYSTEM DIRECTORY OWNER IS [1,4]
377         .WORD   -1,0                    ; INITIALIZE ACL MUTEX
378         .WORD   LNM_SYS_DIR_ORB_SIZ     ; SIZE OF OBJECT RIGHTS BLOCK
379         .BYTE   DYN$C_ORB               ; BLOCK TYPE
380         .BYTE   0                       ; NO ACL AS YET
381         .LONG   0                       ; ZERO RESERVED WORD & REF COUNT
382         .QUAD   0                       ; OBJECT DOES NOT HAVE AN ACCESS MODE
383         .LONG   ^X00000008              ; SYSTEM PROTECTION IS RWE
384         .LONG   ^X00000008              ; OWNER PROTECTION IS RWE
385         .LONG   ^X0000001E              ; GROUP PROTECTION IS R
386         .LONG   ^X0000001E              ; WORLD PROTECTION IS R
387         .LONG   0,0                     ; NULL INITIAL ACL
388         .BYTE   0[ORB$S_MIN_CLASS]      ; MINIMUM CLASSIFICATION MASK
389         .BYTE   0[ORB$S_MAX_CLASS]      ; MAXIMUM CLASSIFICATION MASK
390
391         .ALIGN  5
392 LNM_SYS_DIR_ORB_SIZ = . - LNM_SYSTEM_DIR_ORB
393 LNM_SYS_DIR_SIZ = . - LNM$SYSTEM_DIRECTORY
394
395 ;
396 ; LNM$SYSTEM_TABLE - THE SYSTEM LOGICAL NAME TABLE.
397 ;
398
399 SYSTEM_TABLE:
400         .LONG   0                       ; FORWARD LINK
401         .LONG   0                       ; BACK LINK
402         .WORD   SYSTEM_TABLE_SIZE       ; SIZE OF STRUCTURE
403         .BYTE   DYN$C_LNM               ; TYPE OF STRUCTURE
404         .BYTE   PSL$C_KERNEL            ; KERNEL ACCESS MODE
405         .ADDRESS LNM_SYSTEM_DIR_LNMTH   ; DIRECTORY TABLE HEADER ADDRESS
406         .BYTE   LNMB$M_NO_ALIAS!-       ; TABLE CAN NOT BE ALIASED
407                 LNMB$M_TABLE            ; TABLE
408         .ASCIC  LNM$SYSTEM_TABLE      ; TABLE NAME
409
410         .BYTE   LNMX$M_TERMINAL         ; FLAGS BYTE. NO MORE TRANSLATIONS
411         .BYTE   LNMX$C_TABLE            ; TRANSLATION INDEX ( SPECIAL TABLE )
412         .WORD   0                       ; TRANSLATION HASH CODE
413         .BYTE   LNMTH$K_LENGTH          ; SIZE OF TABLE HEADER BLOCK
414
415 SYSTEM_TABLE_LNMTH:
416         .BYTE   LNMTH$M_SHAREABLE!-     ; TABLE IS SHAREABLE
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 9
X-35U3 DECLARATIONS 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (4)

  417                   LNMTH$M_SYSTEM          ; THIS IS THE SYSTEM LOGICAL NAME TABLE
  418           .LONG   0                       ; ADDRESS OF HASH TABLE
  419           .ADDRESS SYSTEM_TABLE_ORB       ; ADDRESS OF OBJECT RIGHTS BLOCK
  420           .ADDRESS SYSTEM_TABLE           ; ADDRESS OF CONTAINING LNMB BLOCK
  421           .ADDRESS LNM_SYSTEM_DIR_LNMTH   ; ADDRESS OF PARENT TABLE
  422           .LONG   0                       ; ADDRESS OF CHILD TABLE
  423           .LONG   0                       ; ADDRESS OF SIBLING TABLE
  424           .ADDRESS LNM_SYSTEM_DIR_LNMTH   ; ADDRESS OF TABLE HOLDING QUOTA
  425           .LONG   0                       ; INITIAL QUOTA    ( POOLED )
  426           .LONG   0                       ; REMAINING QUOTA ( POOLED )
  427
  428           .BYTE   LNMX$M_XEND             ; FLAGS BYTE. NO MORE TRANSLATIONS
  429
  430           .ALIGN  QUAD
  431 SYSTEM_TABLE_ORB:
  432           .LONG   ^X00010004              ; SYSTEM TABLE OWNER IS [1,4]
  433           .WORD   -1,0                    ; INITIALIZE ACL MUTEX
  434           .WORD   SYSTEM_TABLE_ORB_SIZ    ; SIZE OF OBJECT RIGHTS BLOCK
  435           .BYTE   DYN$C_ORB               ; BLOCK TYPE
  436           .BYTE   0                       ; NO ACL AS YET
  437           .LONG   0                       ; ZERO RESERVED WORD & REF COUNT
  438           .QUAD   0                       ; OBJECT DOES NOT HAVE AN ACCESS MODE
  439           .LONG   ^X00000008              ; SYSTEM PROTECTION IS RWE
  440           .LONG   ^X00000008              ; OWNER PROTECTION IS RWE
  441           .LONG   ^X0000001E              ; GROUP PROTECTION IS R
  442           .LONG   ^X0000001E              ; WORLD PROTECTION IS R
  443           .LONG   0,0                     ; NULL INITIAL ACL
  444           .BYTE   0[ORB$S_MIN_CLASS]      ; MINIMUM CLASSIFICATION MASK
  445           .BYTE   0[ORB$S_MAX_CLASS]      ; MAXIMUM CLASSIFICATION MASK
  446
  447           .ALIGN  5
  448 SYSTEM_TABLE_ORB_SIZ = . - SYSTEM_TABLE_ORB
  449 SYSTEM_TABLE_SIZE = . - SYSTEM_TABLE
```

```
451
452         DECLARE_PSECT   EXEC$PAGED_DATA
453
454 IMGDESC:.ASCID  /SYSINIT.EXE/           ; SYSTEM INITIALIZATION PROCESS
455 TTODESC:.ASCID  /OPA0:/
456
457 ;
458 ; DESCRIPTORS AND CHARACTER STRING BUFFERS FOR THE LOGICAL NAME TABLE NAMES,
459 ; LOGICAL NAMES, AND LOGICAL NAME EQUIVALENCE STRINGS THAT ARE CREATED AT
460 ; SYSTEM INITIALIZATION TIME.
461 ;
462
463 LNM_DIRECTORIES_DESC:
464         .ASCID  /LNM$DIRECTORIES/
465
466 LNM_FILE_DEV_DESC:
467         .ASCID  /LNM$FILE_DEV/
468
469 LNM_PERMANENT_MAILBOX_DESC:
470         .ASCID  /LNM$PERMANENT_MAILBOX/

471
472 LNM_SYSTEM_DESC:
473         .LONG   LNM_SYSTEM_LENGTH
474         .ADDRESS LNM_SYSTEM
475
476 LNM_SYSTEM_DIRECTORY_DESC:
477         .LONG   LNM_SYSTEM_DIRECTORY_LENGTH
478         .ADDRESS LNM_SYSTEM_DIRECTORY
479
480 LNM_TEMPORARY_MAILBOX_DESC:
481         .ASCID  /LNM$TEMPORARY_MAILBOX/

482
483 LOG_G_DESC:
484         .LONG   LOG_GROUP_LENGTH
485         .ADDRESS LOG_GROUP
486
487 LOG_P_DESC:
488         .LONG   LOG_PROCESS_LENGTH
489         .ADDRESS LOG_PROCESS
490
491 LOG_S_DESC:
492         .LONG   LOG_SYSTEM_LENGTH
493         .ADDRESS LOG_SYSTEM
494
495 SYS_DISK_DESC:
496         .ASCID  /SYS$DISK/
497
498 SYS_SYSDEVICE_DESC:
499         .ASCID  /SYS$SYSDEVICE/
```

```
  500
  501 TRNLOG_GS_DESC:
  502         .ASCID  /TRNLOG$_GROUP_SYSTEM/

  503
  504 TRNLOG_PG_DESC:
  505         .ASCID  /TRNLOG$_PROCESS_GROUP/

  506
  507 TRNLOG_PS_DESC:
  508         .ASCID  /TRNLOG$_PROCESS_SYSTEM/

  509
  510 TRNLOG_PGS_DESC:
  511         .ASCID  /TRNLOG$_PROCESS_GROUP_SYSTEM/

  512
  513 LNM_GROUP:
  514         .ASCII  /LNM$GROUP/
  515 LNM_GROUP_LENGTH = . - LNM_GROUP
  516
  517 LNM_JOB:
  518         .ASCII  /LNM$JOB/
  519 LNM_JOB_LENGTH  = . - LNM_JOB
  520
  521 LNM_PROCESS:
  522         .ASCII  /LNM$PROCESS/
  523 LNM_PROCESS_LENGTH = . - LNM_PROCESS
  524
  525 LNM_PROCESS_DIRECTORY:
  526         .ASCII  /LNM$PROCESS_DIRECTORY/
  527 LNM_PROCESS_DIRECTORY_LENGTH = . - LNM_PROCESS_DIRECTORY
  528
  529 LNM_SYSTEM:
  530         .ASCII  /LNM$SYSTEM/
  531 LNM_SYSTEM_LENGTH = . - LNM_SYSTEM
  532
  533 LNM_SYSTEM_DIRECTORY:
  534         .ASCII  /LNM$SYSTEM_DIRECTORY/
  535 LNM_SYSTEM_DIRECTORY_LENGTH = . - LNM_SYSTEM_DIRECTORY
  536
  537 LNM_SYSTEM_TABLE:
  538         .ASCII  /LNM$SYSTEM_TABLE/
  539 LNM_SYSTEM_TABLE_LENGTH = . - LNM_SYSTEM_TABLE
  540
  541 LOG_GROUP:
  542         .ASCII  /LOG$GROUP/
  543 LOG_GROUP_LENGTH = . - LOG_GROUP
  544
```

```
545 LOG_PROCESS:
546         .ASCII  /LOG$PROCESS/
547 LOG_PROCESS_LENGTH = . - LOG_PROCESS
548
549 LOG_SYSTEM:
550         .ASCII  /LOG$SYSTEM/
551 LOG_SYSTEM_LENGTH = . - LOG_SYSTEM
552
553 ;
554 ; ATTRIBUTE, ACCESS MODE AND ITEM BUFFERS WHICH ARE PASSED BY REFERENCE.
555 ;
556
557 EXEC_MODE:                              ; EXECUTIVE ACCESS MODE BUFFER
558         .LONG   PSL$C_EXEC
559
560 KERNEL_MODE:                            ; KERNEL ACCESS MODE BUFFER
561         .LONG   PSL$C_KERNEL
562
563 SUPER_MODE:                             ; SUPERVISOR ACCESS MODE BUFFER
564         .LONG   PSL$C_SUPER
565
566 LNM_NO_ALIAS:                           ; NO_ALIAS ATTRIBUTE BUFFER
567         .LONG   LNM$M_NO_ALIAS
568
569 TERMINAL_BUFFER:                        ; TERMINAL ATTRIBUTES ITEM BUFFER
570         .LONG   LNM$M_TERMINAL
571
572 ;
573 ; ITEM LISTS FOR THE CREATION OF THE LOGICAL NAMES SETUP AT SYSTEM
574 ; INITIALIZATION TIME.
575 ;
576
577 DIRECTORIES_LIST:                       ; ITEM LIST FOR LNM$DIRECTORIES
578         .WORD   4                       ; TERMINAL ATTRIBUTES ITEM
579         .WORD   LNM$_ATTRIBUTES
580         .ADDRESS TERMINAL_BUFFER
581         .LONG   0
582
583         .WORD   LNM_PROCESS_DIRECTORY_LENGTH ; LNM$PROCESS_DIRECTORY STRING ITEM
584         .WORD   LNM$_STRING
585         .ADDRESS LNM_PROCESS_DIRECTORY
586         .LONG   0
587
588         .WORD   LNM_SYSTEM_DIRECTORY_LENGTH ; LNM$SYSTEM_DIRECTORY STRING ITEM
589         .WORD   LNM$_STRING
590         .ADDRESS LNM_SYSTEM_DIRECTORY
591         .LONG   0
592
593         .LONG   0                       ; END OF ITEM LIST
594
595 FILE_DEV_SUPER_LIST:                    ; ITEM LIST FOR SUPERVISOR LNM$FILE_DEV
596         .WORD   LNM_PROCESS_LENGTH      ; LNM$PROCESS STRING ITEM
597         .WORD   LNM$_STRING
598         .ADDRESS LNM_PROCESS
599         .LONG   0
600
601         .WORD   LNM_JOB_LENGTH          ; LNM$JOB STRING ITEM
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 13
X-35U3 DECLARATIONS 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (6)

  602          .WORD   LNM$_STRING
  603          .ADDRESS LNM_JOB
  604          .LONG   0
  605
  606          .WORD   LNM_GROUP_LENGTH        ; LNM$GROUP STRING ITEM
  607          .WORD   LNM$_STRING
  608          .ADDRESS LNM_GROUP
  609          .LONG   0
  610
  611          .WORD   LNM_SYSTEM_LENGTH       ; LNM$SYSTEM STRING ITEM
  612          .WORD   LNM$_STRING
  613          .ADDRESS LNM_SYSTEM
  614          .LONG   0
  615
  616          .LONG   0                       ; END OF ITEM LIST
  617
  618 FILE_DEV_EXEC_LIST:                      ; ITEM LIST FOR EXECUTIVE LNM$FILE_DEV
  619 PERMANENT_MAILBOX_LIST:                  ; ITEM LIST FOR LNM$PERMANENT_MAILBOX
  620          .WORD   LNM_SYSTEM_LENGTH       ; LNM$SYSTEM STRING ITEM
  621          .WORD   LNM$_STRING
  622          .ADDRESS LNM_SYSTEM
  623          .LONG   0
  624
  625          .LONG   0                       ; END OF ITEM LIST
  626
  627 LOG_G_LIST:                              ; ITEM LIST FOR LOG$GROUP
  628          .WORD   LNM_GROUP_LENGTH        ; LNM$GROUP STRING ITEM
  629          .WORD   LNM$_STRING
  630          .ADDRESS LNM_GROUP
  631          .LONG   0
  632
  633          .LONG   0                       ; END OF ITEM LIST
  634
  635 LOG_P_LIST:                              ; ITEM LIST FOR LOG$PROCESS
  636          .WORD   LNM_PROCESS_LENGTH      ; LNM$PROCESS STRING ITEM
  637          .WORD   LNM$_STRING
  638          .ADDRESS LNM_PROCESS
  639          .LONG   0
  640
  641          .WORD   LNM_JOB_LENGTH          ; LNM$JOB STRING ITEM
  642          .WORD   LNM$_STRING
  643          .ADDRESS LNM_JOB
  644          .LONG   0
  645
  646          .LONG   0                       ; END OF ITEM LIST
  647
  648 LOG_S_LIST:                              ; ITEM LIST FOR LOG$SYSTEM
  649          .WORD   LNM_SYSTEM_LENGTH       ; LNM$SYSTEM STRING ITEM
  650          .WORD   LNM$_STRING
  651          .ADDRESS LNM_SYSTEM
  652          .LONG   0
  653
  654          .LONG   0                       ; END OF ITEM LIST
  655
  656 SYSTEM_LIST:                             ; ITEM LIST FOR LNM$SYSTEM
  657          .WORD   4                       ; TERMINAL ATTRIBUTES ITEM
  658          .WORD   LNM$_ATTRIBUTES
```

```
659          .ADDRESS TERMINAL_BUFFER
660          .LONG    0
661
662          .WORD    LNM_SYSTEM_TABLE_LENGTH ; LNM$SYSTEM_TABLE STRING ITEM
663          .WORD    LNM$_STRING
664          .ADDRESS LNM_SYSTEM_TABLE
665          .LONG    0
666
667          .LONG    0                       ; END OF ITEM LIST
668
669 TEMPORARY_MAILBOX_LIST:                    ; ITEM LIST FOR LNM$TEMPORARY_MAILBOX
670          .WORD    LNM_JOB_LENGTH          ; LNM$JOB STRING ITEM
671          .WORD    LNM$_STRING
672          .ADDRESS LNM_JOB
673          .LONG    0
674
675          .LONG    0                       ; END OF ITEM LIST
676
677 TRNLOG_GS_LIST:                           ; ITEM LIST FOR TRNLOG$_GROUP_SYSTEM
678          .WORD    LOG_GROUP_LENGTH        ; LOG$GROUP STRING ITEM
679          .WORD    LNM$_STRING
680          .ADDRESS LOG_GROUP
681          .LONG    0
682
683          .WORD    LOG_SYSTEM_LENGTH       ; LOG$SYSTEM STRING ITEM
684          .WORD    LNM$_STRING
685          .ADDRESS LOG_SYSTEM
686          .LONG    0
687
688          .LONG    0                       ; END OF ITEM LIST
689
690 TRNLOG_PG_LIST:                           ; ITEM LIST FOR TRNLOG$_PROCESS_GROUP
691          .WORD    LOG_PROCESS_LENGTH      ; LOG$PROCESS STRING ITEM
692          .WORD    LNM$_STRING
693          .ADDRESS LOG_PROCESS
694          .LONG    0
695
696          .WORD    LOG_GROUP_LENGTH        ; LOG$GROUP STRING ITEM
697          .WORD    LNM$_STRING
698          .ADDRESS LOG_GROUP
699          .LONG    0
700
701          .LONG    0                       ; END OF ITEM LIST
702
703 TRNLOG_PS_LIST:                           ; ITEM LIST FOR TRNLOG$_PROCESS_SYSTEM
704          .WORD    LOG_PROCESS_LENGTH      ; LOG$PROCESS STRING ITEM
705          .WORD    LNM$_STRING
706          .ADDRESS LOG_PROCESS
707          .LONG    0
708
709          .WORD    LOG_SYSTEM_LENGTH       ; LOG$SYSTEM STRING ITEM
710          .WORD    LNM$_STRING
711          .ADDRESS LOG_SYSTEM
712          .LONG    0
713
714          .LONG    0                       ; END OF ITEM LIST
715
```

```
  716 TRNLOG_PGS_LIST:                        ; ITEM LIST FOR TRNLOG$_PROCESS_GROUP_SYSTEM
  717         .WORD   LOG_PROCESS_LENGTH      ; LOG$PROCESS STRING ITEM
  718         .WORD   LNM$_STRING
  719         .ADDRESS LOG_PROCESS
  720         .LONG   0
  721
  722         .WORD   LOG_GROUP_LENGTH        ; LOG$GROUP STRING ITEM
  723         .WORD   LNM$_STRING
  724         .ADDRESS LOG_GROUP
  725         .LONG   0
  726
  727         .WORD   LOG_SYSTEM_LENGTH       ; LOG$SYSTEM STRING ITEM
  728         .WORD   LNM$_STRING
  729         .ADDRESS LOG_SYSTEM
  730         .LONG   0
  731
  732         .LONG   0                       ; END OF ITEM LIST
  733
  734 ;
  735 ; ARGUMENT LISTS FOR THE $CRELNMS. THIS SYSTEM SERVICES CAN NOT BE DIRECTLY
  736 ; ISSUED AT SYSTEM INITIALIZATION BECAUSE THE SWAPPER DOES NOT HAVE A P1 SPACE
  737 ; WITH SYSTEM SERVICE VECTORS; HOWEVER, IT MAYBE CALLED DIRECTLY. SETUP AN
  738 ; ARGUMENT LIST FOR EACH AND EVERY DIRECT CALL.
  739 ;
  740
  741 DIRECTORIES_ARG:                        ; ARGUMENT LIST FOR LNM$DIRECTORIES
  742         $CRELNM -
  743                 ACMODE  = KERNEL_MODE, -
  744                 ATTR    = LNM_NO_ALIAS, -
  745                 ITMLST  = DIRECTORIES_LIST, -
  746                 LOGNAM  = LNM_DIRECTORIES_DESC, -
  747                 TABNAM  = LNM_SYSTEM_DIRECTORY_DESC
  748
  749 FILE_DEV_EXEC_ARG:                     ; ARGUMENT LIST FOR EXECUTIVE LNM$FILE_DEV
  750         $CRELNM -
  751                 ACMODE  = EXEC_MODE, -
  752                 ITMLST  = FILE_DEV_EXEC_LIST, -
  753                 LOGNAM  = LNM_FILE_DEV_DESC, -
  754                 TABNAM  = LNM_SYSTEM_DIRECTORY_DESC
  755
  756 FILE_DEV_SUPER_ARG:                     ; ARGUMENT LIST FOR SUPERVISOR LNM$FILE_DEV
  757         $CRELNM -
  758                 ACMODE  = SUPER_MODE, -
  759                 ITMLST  = FILE_DEV_SUPER_LIST, -
  760                 LOGNAM  = LNM_FILE_DEV_DESC, -
  761                 TABNAM  = LNM_SYSTEM_DIRECTORY_DESC
  762
  763 LOG_G_ARG:                              ; ARGUMENT LIST FOR LOG$GROUP
  764         $CRELNM -
  765                 ACMODE  = KERNEL_MODE, -
  766                 ITMLST  = LOG_G_LIST, -
  767                 LOGNAM  = LOG_G_DESC, -
  768                 TABNAM  = LNM_SYSTEM_DIRECTORY_DESC
  769
  770 LOG_P_ARG:                              ; ARGUMENT LIST FOR LOG$PROCESS
  771         $CRELNM -
  772                 ACMODE  = KERNEL_MODE, -
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 16
X-35U3 DECLARATIONS 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (6)

                ITMLST  = LOG_P_LIST, -
                LOGNAM  = LOG_P_DESC, -
                TABNAM  = LNM_SYSTEM_DIRECTORY_DESC

LOG_S_ARG:                              ; ARGUMENT LIST FOR LOG$SYSTEM
        $CRELNM -
                ACMODE  = KERNEL_MODE, -
                ITMLST  = LOG_S_LIST, -
                LOGNAM  = LOG_S_DESC, -
                TABNAM  = LNM_SYSTEM_DIRECTORY_DESC

PERMANENT_MAILBOX_ARG:                  ; ARGUMENT LIST FOR LNM$PERMANENT_MAILBOX
        $CRELNM -
                ACMODE  = KERNEL_MODE, -
                ITMLST  = PERMANENT_MAILBOX_LIST, -
                LOGNAM  = LNM_PERMANENT_MAILBOX_DESC, -
                TABNAM  = LNM_SYSTEM_DIRECTORY_DESC

SYSTEM_ARG:                             ; ARGUMENT LIST FOR LNM$SYSTEM
        $CRELNM -
                ACMODE  = KERNEL_MODE, -
                ATTR    = LNM_NO_ALIAS, -
                ITMLST  = SYSTEM_LIST, -
                LOGNAM  = LNM_SYSTEM_DESC, -
                TABNAM  = LNM_SYSTEM_DIRECTORY_DESC

TEMPORARY_MAILBOX_ARG:                  ; ARGUMENT LIST FOR LNM$TEMPORARY_MAILBOX
        $CRELNM -
                ACMODE  = KERNEL_MODE, -
                ITMLST  = TEMPORARY_MAILBOX_LIST, -
                LOGNAM  = LNM_TEMPORARY_MAILBOX_DESC, -
                TABNAM  = LNM_SYSTEM_DIRECTORY_DESC

TRNLOG_GS_ARG:                          ; ARGUMENT LIST FOR TRNLOG$_GROUP_SYSTEM
        $CRELNM -
                ACMODE  = KERNEL_MODE, -
                ITMLST  = TRNLOG_GS_LIST, -
                LOGNAM  = TRNLOG_GS_DESC, -
                TABNAM  = LNM_SYSTEM_DIRECTORY_DESC

TRNLOG_PG_ARG:                          ; ARGUMENT LIST FOR TRNLOG$_PROCESS_GROUP
        $CRELNM -
                ACMODE  = KERNEL_MODE, -
                ITMLST  = TRNLOG_PG_LIST, -
                LOGNAM  = TRNLOG_PG_DESC, -
                TABNAM  = LNM_SYSTEM_DIRECTORY_DESC

TRNLOG_PS_ARG:                          ; ARGUMENT LIST FOR TRNLOG$_PROCESS_SYSTEM
        $CRELNM -
                ACMODE  = KERNEL_MODE, -
                ITMLST  = TRNLOG_PS_LIST, -
                LOGNAM  = TRNLOG_PS_DESC, -
                TABNAM  = LNM_SYSTEM_DIRECTORY_DESC

TRNLOG_PGS_ARG:                         ; ARGUMENT LIST FOR TRNLOG$_PROCESS_GROUP_SY
        $CRELNM -
                ACMODE  = KERNEL_MODE, -
```

```
                ITMLST  = TRNLOG_PGS_LIST, -
                LOGNAM  = TRNLOG_PGS_DESC, -
                TABNAM  = LNM_SYSTEM_DIRECTORY_DESC

        DECLARE_PSECT  EXEC$NONPAGED_DATA,ALIGNMENT=PAGE
                                        ; WRITABLE PSECT
                                        ; ITMLST MUST BE FOLLOWING TWO CRELNM

SYS_DISK_ARG:                           ; ARGUMENT LIST FOR SYS$DISK
        $CRELNM -
                ACMODE  = EXEC_MODE, -
                LOGNAM  = SYS_DISK_DESC, -
                TABNAM  = LNM_SYSTEM_DESC

SYS_SYSDEVICE_ARG:                      ; ARGUMENT LIST FOR SYS$SYSDEVICE
        $CRELNM -
                ACMODE  = EXEC_MODE, -
                LOGNAM  = SYS_SYSDEVICE_DESC, -
                TABNAM  = LNM_SYSTEM_DESC
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 18
X-35U3 DECLARATIONS 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (8)

                        DECLARE_PSECT   EXEC$PAGED_DATA

;
; DEFINE A QUOTA LIST TO BE USED BY VARIOUS PIECES OF THE SYSTEM WHEN
; CREATING A SPECIAL SYSTEM PROCESS, LIKE A FILES-11 ACP.  EVERY QUOTA
; IS MENTIONED EXPLICITLY.  NOTE THAT THIS LIST CAN BE TAILORED BY
; COPYING IT TO SOME TEMPORARY LOCATION AND APPENDING NEW QUOTA ITEMS
; TO THE END OF THE LIST.  THE $CREPRC SYSTEM SERVICE USES THE LAST
; VALUE OF A SPECIFIED QUOTA IN THE LIST WHEN IT CREATES A PROCESS.
; NOTE THAT THE END OF THE LIST MUST BE TERMINATED BY A ZERO BYTE,
; AND THAT THE LENGTH OF THE LIST, AS GIVEN BY PQL$C_SYSPQLLEN, DOES NOT
; INCLUDE THE LIST TERMINATOR.
;

        UNIVERSAL_SYMBOL        PQL$AR_SYSPQL
                                        ; This is the name of the pointer
                                        ; used by other modules
PQL$AB_SYSPQL:                          ; This is the name used by SWAPPER
                                        ; SYSTEM PROCESS QUOTA LIST
        .BYTE   PQL$_ASTLM              ; PROCESS AST LIMIT
        .LONG   10
        .BYTE   PQL$_BIOLM              ; PROCESS BUFFERED I/O LIMIT
        .LONG   10
        .BYTE   PQL$_BYTLM              ; PROCESS BUFFERED I/O BYTE LIMIT
        .LONG   32768
        .BYTE   PQL$_CPULM              ; PROCESS CPU TIME LIMIT
        .LONG   0                       ; ZERO IMPLIES NO LIMIT
        .BYTE   PQL$_DIOLM              ; PROCESS DIRECT I/O LIMIT
        .LONG   10
        .BYTE   PQL$_FILLM              ; PROCESS OPEN FILE LIMIT
        .LONG   60
        .BYTE   PQL$_PGFLQUOTA          ; PROCESS PAGE FILE QUOTA
        .LONG   20000
        .BYTE   PQL$_PRCLM              ; PROCESS SUBPROCESS CREATION LIMIT
        .LONG   8
        .BYTE   PQL$_TQELM              ; PROCESS TIMER QUEUE ENTRY LIMIT
        .LONG   8
        .BYTE   PQL$_WSDEFAULT          ; PROCESS DEFAULT WORKING SET SIZE
        .LONG   100
        .BYTE   PQL$_WSQUOTA            ; PROCESS WORKING SET QUOTA
        .LONG   200
        .BYTE   PQL$_WSEXTENT           ; PROCESS WORKING SET EXTENT LIMIT
        .LONG   1000
        .BYTE   PQL$_ENQLM              ; PROCESS LOCK LIMIT
        .LONG   100
        .BYTE   PQL$_JTQUOTA            ; JOB-WIDE LOGICAL NAME TABLE QUOTA
        .LONG   1024
10$:    .BYTE   PQL$_LISTEND            ; END OF PROCESS QUOTA LIST

;
; The length of the SYSPQL list (given by the constant PQL$C_SYSPQLLEN)
; will be stuffed into the PQL$GL_SYSPQLLEN cell, which resides in the base
; image, by the swapper's initialization routine.
;

PQL$C_SYSPQLLEN == 10$ - PQL$AB_SYSPQL  ; LENGTH OF LIST (MINUS TERMINATOR)
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 19
X-35U3 DECLARATIONS 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (8)

  907
  908
  909 PQL$AB_SYSPQL_ALT:                      ; PQL list used to run stand-alone
  910                                         ; application (typically, SA BACKUP)
  911         .BYTE   PQL$_ASTLM              ; PROCESS AST LIMIT
  912         .LONG   <<1@15>-1>
  913         .BYTE   PQL$_BIOLM              ; PROCESS BUFFERED I/O LIMIT
  914         .LONG   <<1@15>-1>
  915         .BYTE   PQL$_BYTLM              ; PROCESS BUFFERED I/O BYTE LIMIT
  916         .LONG   <<1@15>-1>
  917         .BYTE   PQL$_CPULM              ; PROCESS CPU TIME LIMIT
  918         .LONG   0                       ; ZERO IMPLIES NO LIMIT
  919         .BYTE   PQL$_DIOLM              ; PROCESS DIRECT I/O LIMIT
  920         .LONG   <<1@15>-1>
  921         .BYTE   PQL$_FILLM              ; PROCESS OPEN FILE LIMIT
  922         .LONG   <<1@15>-1>
  923         .BYTE   PQL$_PGFLQUOTA          ; PROCESS PAGE FILE QUOTA
  924         .LONG   <<<1@31>-1>/512>
  925         .BYTE   PQL$_PRCLM              ; PROCESS SUBPROCESS CREATION LIMIT
  926         .LONG   <<1@15>-1>
  927         .BYTE   PQL$_TQELM              ; PROCESS TIMER QUEUE ENTRY LIMIT
  928         .LONG   <<1@15>-1>
  929
  930         ASSUME  PTE$S_PFN GE 16
  931         .BYTE   PQL$_WSDEFAULT          ; PROCESS DEFAULT WORKING SET SIZE
  932         .LONG   <<1@16>-1>
  933         .BYTE   PQL$_WSQUOTA            ; PROCESS WORKING SET QUOTA
  934         .LONG   <<1@16>-1>
  935
  936         .BYTE   PQL$_WSEXTENT           ; PROCESS WORKING SET EXTENT LIMIT
  937         .LONG   <<1@PTE$S_PFN>-1>
  938         .BYTE   PQL$_ENQLM              ; PROCESS LOCK LIMIT
  939         .LONG   <<1@15>-1>
  940         .BYTE   PQL$_JTQUOTA            ; JOB-WIDE LOGICAL NAME TABLE QUOTA
  941         .LONG   <<1@15>-1>
  942         .BYTE   PQL$_LISTEND            ; END OF PROCESS QUOTA LIST
  943
```

```
  945          .SBTTL  EXE$SWAPINIT - INITIALIZATION AND STARTUP FOR SWAPPER
  946
  947 ;++
  948 ; FUNCTIONAL DESCRIPTION:
  949 ;       EXE$SWAPINIT IS ENTERED WHEN THE SWAPPER PROCESS IS FIRST
  950 ;       SCHEDULED AFTER A SYSTEM BOOT/STARTUP.  THIS TRANSFER OCCURS
  951 ;       VIA THE INITIAL PC VALUE BUILT INTO THE HARDWARE PCB FOR THE
  952 ;       SWAPPER PROCESS.  R4 CONTAINS THE ADDRESS OF THE SWAPPER PCB.
  953 ;
  954 ;--
  955          DECLARE_PSECT   EXEC$PAGED_CODE
  956
  957          UNIVERSAL_SYMBOL        EXE$SWAPINIT
  958 ;EXE$SWAPINIT::                          ; SWAPPER INITIALIZATION
  959          BBSS    #BOOSTATE$V_SWAPPER,-   ; INDICATE THAT WE'RE IN THE SWAPPER
  960                  G^EXE$GL_STATE,1$
  961 1$:      CLRL    PCB$L_WSSWP(R4)         ; Initialize mapping window for
  962          MOVL    #<<1@31>!SWP$C_SHELLSIZ>,- ; SHELL read I/O
  963                  PCB$L_SWAPSIZE(R4)      ; (Bit 31 is END OF WINDOW flag)
  964 ;
  965 ; INITIALIZE PAGED POOL.
  966 ;
  967          MOVL    G^EXE$GL_PAGED,R11      ; POINT TO START OF PAGED POOL
  968          CLRL    (R11)+                  ; ZAP FORWARD LINK
  969          MOVL    G^SGN$GL_PAGEDYN,(R11)  ; AND SET SIZE
  970
  971 ;
  972 ; IF THE SYSTEM DISK IS A SHADOW SET, WAIT FOR THE DISK CLASS DRIVER TO
  973 ; CREATE THE SHADOW SET BEFORE INITIALIZING LOGICAL NAME DATA STRUCTURES.
  974 ;
  975 2$:      MOVL    G^EXE$GL_RPB,R0         ; Get RPB address.
  976          MOVW    RPB$L_BOOTR3+2(R0),R3   ; Get high word of boot R3.
  977          BGEQ    4$                      ; If GEQ, not shadowed system disk.
  978          MOVL    G^EXE$GL_SYSUCB,R1      ; Get system disk's UCB address.
  979          CMPW    UCB$W_MSCPUNIT(R1),R3   ; Is the system disk a shadow set ?
  980          BNEQ    2$                      ; If NEQ, busy loop.
  981 4$:
  982
  983 ;
  984 ; CALL ALL LOADED SYSTEM IMAGE'S INITIALIZATION ROUTINES AGAIN.  THIS IS
  985 ; SO THAT ANY .ADDRESS FIXUPS IN PAGED CODE CAN BE DONE AT THIS TIME.
  986 ;
  987          CALLS   #0,G^LDR$INIT_ALL
  988
  989 ;
  990 ; ALLOCATE LOGICAL NAME HASH TABLE.  THE NUMBER OF ENTRIES IN THE HASH TABLE
  991 ; MUST BE A POWER OF TWO.  SO THE ALLOCATED SIZE IS THE SMALLEST POWER OF
  992 ; TWO LARGER THAN THE SYSGEN PARAMETER.
  993 ;
  994          MOVL    G^LNM$AL_HASHTBL,R8     ; GET POINTER TO HASH TABLE
  995          PUSHL   (R8)                    ; SAVE ADDR OF CRELNM ITMLST BLOCKS FOR
  996                                          ; SYS$DISK AND SYS$SYSDEVICE
  997          MOVL    #1, R8                  ; DO THIS TWICE
  998 40$:     SUBL3   #1,G^LNM$GL_HTBLSIZS[R8],R7 ; PICK UP ONE LESS THAN SYSGEN PARM
  999          CVTLF   R7,R7                   ; CONVERT TO FLOATING
 1000          EXTZV   #7,#7,R7,R7             ; PICK UP EXPONENT-NOW THE POWER OF 2
 1001          CLRL    R1                      ; CLEAR A REGISTER
```

```
1002          BBSS    R7,R1,50$               ; THE SIZE OF THE TABLE ROUNDED UP
1003 50$:     MOVL    R1,G^LNM$GL_HTBLSIZS[R8]; WRITE BACK THE CORRECT VALUE
1004          SOBGEQ  R8,40$                  ; LOOP TWO TIMES
1005 ;
1006 ; INITIALIZE THE SYSTEM SPACE HASH TABLE.
1007 ;
1008          MOVL    G^LNM$GL_HTBLSIZS,R1    ; SIZE OF TABLE IN ENTRIES
1009          MOVAL   @#LNMHSH$K_BUCKET[R1],R1; MULT BY 4 AND ADD OVERHEAD
1010          JSB     G^EXE$ALOPAGED          ; ALLOCATE MEMORY
1011          PUSHR   #^M<R1,R2>              ; SAVE REGISTERS DESTORYED BY MOVC5
1012          MOVC5   #0,#0,#0,R1,(R2)        ; ZERO HASH TABLE
1013          POPR    #^M<R1,R4>              ; RESTORE REGISTERS DESTORYED BY MOVC5
1014                                          ;  NOTE: tHAT R2 COMES BACK AS R4
1015          SUBL3   #1,G^LNM$GL_HTBLSIZS,R0 ; CALC UPPER BOUND OF HASH INDEX
1016          MCOML   R0,LNMHSH$L_MASK(R4)    ; STORE HASH INDEX MASK
1017          MOVW    R1,LNMHSH$W_SIZE(R4)    ; STORE SIZE IN STRUCTURE HEADER
1018          MOVB    #DYN$C_RSHT,LNMHSH$B_TYPE(R4)
1019                                          ; STORE STRUCTURE TYPE
1020          MOVL    G^LNM$AL_HASHTBL,R3     ; POINTER TO BASE OF TABLE
1021          MOVAB   (R4),(R3)               ; STUFF WAY POINTER TO TABLE
1022                                          ; NOTE: THAT THE HASH TABLE HAS BEEN
1023                                          ; INITIALIZED TO ZERO
1024
1025 ;
1026 ; FIX UP THE SYSTEM LOGICAL NAME DIRECTORY, AND INSERT IT IN INTO THE
1027 ; APPROPRIATE HASH BUCKET OF THE SHAREABLE LOGICAL NAME HASH TABLE.
1028 ;
1029
1030          MOVAB   LNM$SYSTEM_DIRECTORY,R3
1031          MOVAL   (R4),LNM_SYSTEM_DIR_LNMTH+LNMTH$L_HASH
1032                                          ; HASH TABLE ADDRESS IN LNMTH
1033          MOVZBL  LNMB$T_NAME(R3),R0      ; GET SIZE OF DIRECTORY NAME
1034          MOVAB   LNMB$T_NAME+1(R3),R1    ; GET ADDRESS OF DIRECTORY NAME
1035          JSB     G^LNM$HASH              ; HASH THE DIRECTORY NAME
1036          BICL2   LNMHSH$L_MASK(R4),R0    ; MODIFY THE INDEX TO BE IN RANGE
1037          MOVAL   (R3),LNMHSH$C_BUCKET(R4)[R0]
1038                                          ; INSERT DIRECTORY LNMB IN HASH TABLE
1039          MOVAL   LNMHSH$C_BUCKET(R4)[R0],LNMB$L_BLINK(R3)
1040                                          ; INSERT HASH TBL ENTRY IN DIRECT LNMB
1041
1042 ;
1043 ; FIXUP THE SYSTEM LOGICAL NAME TABLE, LNM$SYSTEM_TABLE, AND INSERT IT INTO THE
1044 ; APPROPRIATE HASH BUCKET OF THE SYSTEM LOGICAL NAME HASH TABLE.
1045 ;
1046
1047          MOVAB   SYSTEM_TABLE,R1         ; RETRIEVE SYSTEM TABLE LNMB ADDRESS
1048          MOVL    G^LNM$AL_HASHTBL,R2     ; POINTER TO BASE OF TABLE
1049          MOVL    (R2),-                  ; MOVE THE ADDRESS OF THE SHAREABLE
1050                  SYSTEM_TABLE_LNMTH+-    ; LOGICAL NAME HASH TABLE INTO THE
1051                  LNMTH$L_HASH            ; SYSTEM TABLE'S TABLE HEADER
1052          CLRL    R2                      ; NO SPECIAL INSERTION ATTRIBUTES
1053          JSB     G^LNM$INSLOGTAB         ; APPROPRIATELY INSERT LNM$SYSTEM_TABLE
1054          MOVAL   W^LNM$SYSTEM_DIRECTORY,-; Fix up LNM$AL_DIRTBL's pointer to
1055                  G^LNM$AL_DIRTBL+8       ; LNM$SYSTEM_DIRECTORY
1056
1057 ;
1058 ; CREATE THE SYSTEM LOGICAL NAMES, CONTAINED WITHIN THE SYSTEM DIRECTORY TABLE,
```

```
1059 ; - ALL OF WHICH MUST BE CREATED AT SYSTEM INITIALIZATION TIME.
1060 ;
1061
1062         CALLG -                         ; CREATE LNM$DIRECTORIES
1063                 DIRECTORIES_ARG, -
1064                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1065
1066         CALLG -                         ; CREATE EXECUTIVE LNM$FILE_DEV
1067                 FILE_DEV_EXEC_ARG, -
1068                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1069
1070         CALLG -                         ; CREATE SUPERVSIOR LNM$FILE_DEV
1071                 FILE_DEV_SUPER_ARG, -
1072                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1073
1074         CALLG -                         ; CREATE LOG$GROUP
1075                 LOG_G_ARG, -
1076                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1077
1078         CALLG -                         ; CREATE LOG$PROCESS
1079                 LOG_P_ARG, -
1080                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1081
1082         CALLG -                         ; CREATE LOG$SYSTEM
1083                 LOG_S_ARG, -
1084                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1085
1086         CALLG -                         ; CREATE LNM$PERMANENT_MAILBOX
1087                 PERMANENT_MAILBOX_ARG, -
1088                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1089
1090         CALLG -                         ; CREATE LNM$SYSTEM
1091                 SYSTEM_ARG, -
1092                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1093
1094         CALLG -                         ; CREATE LNM$TEMPORARY_MAILBOX
1095                 TEMPORARY_MAILBOX_ARG, -
1096                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1097
1098         CALLG -                         ; CREATE TRNLOG$_GROUP_SYSTEM
1099                 TRNLOG_GS_ARG, -
1100                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1101
1102         CALLG -                         ; CREATE TRNLOG$_PROCESS_GROUP
1103                 TRNLOG_PG_ARG, -
1104                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1105
1106         CALLG -                         ; CREATE TRNLOG$_PROCESS_SYSTEM
1107                 TRNLOG_PS_ARG, -
1108                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1109
1110         CALLG -                         ; CREATE TRNLOG$_PROCESS_GROUP_SYSTEM
1111                 TRNLOG_PGS_ARG, -
1112                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
1113
1114 ;
1115 ; CREATE TWO STARTUP LOGICAL NAMES.
```

```
 1116 ;
 1117
 1118         POPL    R4                      ; RECOVER ADDR OF CRELOG BLOCK
 1119 ;
 1120 ;CRELOG BLOCK CONTENTS:
 1121 ; BYTE 0 - SIZE OF BOTH ITEM LISTS AND ASSOCIATED DATA
 1122 ; BYTE 1 - OFFSET FROM START OF BLOCK TO SECOND ITEM LIST
 1123 ; BYTES 2,3 - PAD BYTES TO LONGWORD ALIGN ITEM LISTS
 1124 ; BYTE 4 - START OF FIRST ITEM LIST FOR $CRELNM
 1125 ;
 1126         MOVAL   4(R4),SYS_DISK_ARG+CRELNM$_ITMLST
 1127                                         ; STUFF THE ADDRESS OF THE ITEM LIST
 1128         MOVZBL  1(R4),R0                ; GET OFFSET TO SYSDEVICE ITEM LIST
 1129         MOVAB   (R4)[R0],SYS_SYSDEVICE_ARG+CRELNM$_ITMLST
 1130                                         ; STUFF THE ADDRESS OF THE ITEM LIST
 1131
 1132         CALLG -                         ; CREATE SYS$DISK
 1133                 SYS_DISK_ARG, -
 1134                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
 1135
 1136         CALLG -                         ; CREATE SYS$SYSDEVICE
 1137                 SYS_SYSDEVICE_ARG, -
 1138                 @#SYS$S0_VECTOR_BASE + BASE$EXE$K_CRELNM
 1139
 1140
 1141         MOVZBL  (R4),R1                 ; GET THE SIZE OF THE STRUCTURE
 1142         MOVL    R4,R0                   ; MOVE STRUCTURE ADDR INTO CORRECT REG
 1143         JSB     G^EXE$DEANONPGDSIZ      ; RETURN THE MEMORY
 1144
 1145 ;
 1146 ; CREATE INITIAL PROCESSES
 1147 ;
 1148
 1149 ; THE $CREPRC_S MACRO CANNOT BE USED BECAUSE THAT MACRO GENERATES A
 1150 ; CALL THROUGH THE P1 SYSTEM SERVICE VECTOR PAGES AND THE SWAPPER DOES
 1151 ; NOT HAVE A P1 SPACE. tHE SENSE OF THE CREATE PROCESS CALL IS THE
 1152 ; FOLLOWING.
 1153 ;
 1154 ;        $CREPRC_S       INPUT=TT0DESC,- ;
 1155 ;                        OUTPUT=TT0DESC,-;
 1156 ;                        ERROR=TT0DESC,- ;
 1157 ;                        IMAGE=IMGDESC,- ;
 1158 ;                        UIC=#^X80020,-  ;
 1159 ;                        STSFLG=#<PRC$M_NOACNT!PRC$M_SSRWAIT>,-
 1160 ;                        BASPRI=#2,-     ;
 1161 ;                        QUOTA=PQL$AB_SYSPQL ;
 1162
 1163         ASSUME  <CREPRC$_STSFLG/4>      EQ 12
 1164         PUSHL   #<PRC$M_NOACNT!PRC$M_SSRWAIT>
 1165
 1166         ASSUME  <CREPRC$_MBXUNT/4>      EQ 11
 1167         CLRL    -(SP)
 1168
 1169         ASSUME  <CREPRC$_UIC/4>         EQ 10
 1170         PUSHL   #^X80020
 1171
 1172         ASSUME  <CREPRC$_BASPRI/4>      EQ  9
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 24
X-35U3 EXE$SWAPINIT - INITIALIZATION AND STARTU 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1

 1173          PUSHL   #2
 1174
 1175          ASSUME  <CREPRC$_PRCNAM/4>      EQ  8
 1176          CLRL    -(SP)
 1177
 1178          ASSUME  <CREPRC$_QUOTA/4>       EQ  7
 1179          PUSHAB  PQL$AB_SYSPQL
 1180          BBC     S^#EXE$V_SA_APP,-       ; If stand-alone application
 1181                  G^EXE$GL_STATIC_FLAGS,60$ ; being booted,
 1182          MOVAB   PQL$AB_SYSPQL_ALT,(SP)  ; use alternate PQL
 1183 60$:
 1184          ASSUME  <CREPRC$_PRVADR/4>      EQ  6
 1185          PUSHL   #0
 1186
 1187          ASSUME  <CREPRC$_ERROR/4>       EQ  5
 1188          PUSHAQ  TTODESC
 1189
 1190          ASSUME  <CREPRC$_OUTPUT/4>      EQ  4
 1191          PUSHL   (SP)
 1192
 1193          ASSUME  <CREPRC$_INPUT/4>       EQ  3
 1194          PUSHL   (SP)
 1195
 1196          ASSUME  <CREPRC$_IMAGE/4>       EQ  2
 1197          PUSHAQ  IMGDESC
 1198
 1199          ASSUME  <CREPRC$_PIDADR/4>      EQ  1
 1200          PUSHL   #0
 1201
 1202          CALLS   #12,@#SYS$SO_VECTOR_BASE + BASE$EXE$K_CREPRC
 1203          JMP     G^LOOP                  ; JUMP OFF TO THE MAIN LOOP
```

```
1206          .SBTTL  SWAPPER - MAIN LOOP
1207
1208 ;++
1209 ; FUNCTIONAL DESCRIPTION:
1210 ;       THE MAIN LOOP OF THE SWAPPER IS EXECUTED WHENEVER THE SWAPPER IS AWAKENED
1211 ;       FOR ANY REASON.  EACH OF THE FUNCTIONAL ROUTINES WILL CHECK TO SEE IF
1212 ;       THEY HAVE ANY ACTION TO PERFORM.
1213 ;--
1214
1215          DECLARE_PSECT   EXEC$NONPAGED_CODE      ; NON-PAGED PSECT
1216 LOOP:    BSBW    BALANCE                 ; BALANCE FREE PAGE COUNT
1217          JSB     G^MMG$WRTMFYPAG         ; WRITE MODIFIED PAGES
1218          BSBW    SWAPSCHED               ; SCHEDULE SWAP
1219          TSTL    G^EXE$GL_PFATIM         ; CHECK FOR POWER FAIL TIME
1220          BEQL    20$                     ; BRANCH IF NO POWERFAIL
1221          SETIPL  #IPL$_ASTDEL            ; Lower IPL
1222          JSB     G^EXE$POWERAST          ; GIVE ANY REQUIRED POWER FAIL ASTS
1223 20$:
1224          FIND_CPU_DATA R4                ; GET THIS CPU'S PER-CPU DATABASE
1225          MOVL    CPU$L_CURPCB(R4),R4     ; GET PROPER PCB ADDRESS
1226          MOVAQ   G^SCH$GQ_HIBWQ,R2       ; AND ADDRESS OF WAIT QUEUE HEADER
1227
1228          ; Lock SCHED database
1229
1230          LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
1231                  PRESERVE=NO             ; OK to destroy R0
1232          BBSC    #PCB$V_WAKEPEN,PCB$L_STS(R4),40$ ; TEST AND CLEAR WAKE PENDING
1233          PUSHL   #0                      ; NULL PSL
1234          JSB     G^SCH$WAITK             ; WAIT WITH STACK CLEAN
1235          BRW     LOOP                    ; CHECK FOR WORK TO DO (ON WAKEUP)
1236
1237 40$:    UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATABASE
1238                  NEWIPL=#0,-             ; LOWER IPL
1239                  PRESERVE=NO             ; OK to destroy R0
1240          BRW     LOOP                    ; CHECK FOR WORK TO DO
1241          .DISABLE LSB
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 26
X-35U3 BALANCE FREE PAGE COUNT 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (11)

 1244          .SBTTL  BALANCE FREE PAGE COUNT
 1245
 1246 ;++
 1247 ; FUNCTIONAL DESCRIPTION:
 1248 ;       BALANCE WILL ENSURE THAT THE FREE PAGE LIST HAS AT LEAST THE NUMBER OF
 1249 ;       PAGES SPECIFIED BY THE PARAMETER FREELIM.  IF NOT, PAGES WILL BE MADE
 1250 ;       AVAILABLE BY EITHER WRITING MODIFIED PAGES OR OUTSWAPPING PROCESSES.
 1251 ;       IF SUFFICIENT FREE PAGES ARE AVAILABLE, THEN A CHECK IS MADE FOR
 1252 ;       DELETED PROCESS HEADERS IN NEED OF CLEANUP.
 1253 ;
 1254 ; ENVIRONMENT:
 1255 ;       ENTERED AT IPL = ASTDEL
 1256 ;--
 1257
 1258 BALANCE:                                    ; BALANCE FREE PAGE COUNT
 1259          LOCK    LOCKNAME=MMG,-            ; LOCK MMG DATABASE
 1260                  PRESERVE=NO               ; OK to destroy R0
 1261          LOCK    LOCKNAME=SCHED,-          ; LOCK SCHED DATABASE
 1262                  CONDITION=NOSETIPL,-      ; (WELL, DON'T REALLY SET THE IPL)
 1263                  PRESERVE=NO               ; OK to destroy R0
 1264          CMPL    G^SGN$GL_FREELIM,G^SCH$GL_FREECNT ; ARE WE HERE DUE TO FREELIM?
 1265          BLEQ    10$                       ; BRANCH IF NOT
 1266          SUBL3   G^SGN$GL_FREEGOAL,G^SCH$GL_FREECNT,R3 ; SUFFICIENT FREE PAGES?
 1267          BLSS    40$                       ; NO, MUST ACQUIRE SOME
 1268 10$:     TSTW    G^SCH$GW_DELPHDCT         ; CHECK FOR DELETED PROCESS HEADERS
 1269          BNEQ    50$                       ; FOUND SOME, FREE THEM UP
 1270
 1271 20$:     ; Nothing to do - unlock MMG and SCHED databases
 1272
 1273          UNLOCK  LOCKNAME=MMG,-            ; UNLOCK MMG DATABASE, LEAVE IPL
 1274                  PRESERVE=NO               ; OK to destroy R0
 1275          UNLOCK  LOCKNAME=SCHED,-          ; UNLOCK SCHED DATABASE
 1276                  NEWIPL=#IPL$_ASTDEL,-     ; LOWER IPL
 1277                  PRESERVE=NO               ; OK to destroy R0
 1278          RSB                               ; IN BALANCE, RETURN
 1279
 1280
 1281 40$:     ; Insufficient free pages - try to find some on the modified page
 1282          ; list.
 1283          ;
 1284          ; IPL = SYNCH, MMG & SCHED SPINLOCKS HELD
 1285
 1286          BBS     S^#SCH$V_MPW,G^SCH$GB_SIP,45$   ; MODIFIED PAGE WRITING ACTIVE
 1287          CMPL    G^SCH$GL_MFYCNT,G^MPW$GL_THRESH ; Enough pages to consider ?
 1288          BLEQ    60$                       ; No, must outswap
 1289
 1290 ; NB: The use of SCH$GL_MFYLOSV instead of SCH$GL_MFYLOLIM is intentional.
 1291 ;     The former is the static copy of the SYSGEN parameter MPW_LOLIMIT,
 1292 ;     while the latter is dynamically changed to control modified page writing.
 1293 ;
 1294
 1295          SUBL3   G^SCH$GL_MFYLOSV,G^SCH$GL_MFYCNT,R0 ; HOW MUCH WILL WRITING PAGES
 1296          BLEQ    60$                       ; NONE, MUST OUTSWAP
 1297          ADDL    R3,R0                     ; YIELD RELATIVE TO WHAT WE NEED?
 1298          BLSS    60$                       ; NOT ENOUGH, MUST OUTSWAP
 1299          MOVL    #<MPW$C_MAINTAIN!MPW$M_LOLIMIT>,R0 ; MAINTAINence purge request
 1300          ADDL3   R3,G^SCH$GL_MFYCNT,R1     ; New MPL low limit
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 27
X-35U3 BALANCE FREE PAGE COUNT 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (11)

 1301          JSB     G^MMG$PURGEMPL          ; Initiate modified page writing
 1302          BRB     20$                     ; AND EXIT TO LET IT HAPPEN
 1303
 1304 45$:     ADDL3   R3,G^MPW$GL_IOPAGCNT,R0 ; Will modified pages in transit help ?
 1305          BLSS    60$                     ; No, must outswap
 1306          BRB     10$                     ; Yes, check for deleted process headers
 1307
 1308 ;
 1309 ;       MUST OUTSWAP, FIRST CHECK FOR SWAP IN PROGRESS SINCE SWAPPER IS
 1310 ;       NOT RE-ENTRANT.  IF PURGING DELETED HEADERS, THEN THE NUMBER OF
 1311 ;       REQUIRED PAGES (IN R3) WILL BE SET TO ZERO.  AN INFINITE INSWAP PRIORITY
 1312 ;       WILL BE ASSUMED WHILE BALANCING THE NUMBER FO FREE PAGES.
 1313 ;
 1314 ;       IPL = SYNCH, MMG & SCHED SPINLOCKS HELD
 1315 ;
 1316 50$:     CLRL    R3                      ;NO FREE PAGES NEEDED, WE'RE JUST
 1317                                          ;DELETING PHD PAGES
 1318 60$:     BBSS    S^#SCH$V_SIP,G^SCH$GB_SIP,20$   ; EXIT IF SWAPPER ALREADY BUSY
 1319
 1320          ;
 1321          ; Perform OUTSWAP operation
 1322          ;
 1323          PUSHAB  B^20$                   ; Push return address
 1324                                          ; - A cheap subroutine call
 1325
 1326          PUSHR   #^M<R6,R7,R8,R9,R10,R11,AP,FP>  ; SAVE NON-STANDARD REGISTERS
 1327          CLRB    G^SWP$GB_ISWPRI         ; SET PRIORITY FOR SWAP SCHEDULE
 1328          MOVL    R3,FP                   ; GET AND TEST FREE PAGE DEFICIT
 1329          BGEQ    80$                     ; NONE, PURGING DELETED HEADERS
 1330          TSTW    G^SWP$GW_BALCNT         ; CHECK FOR SINGULAR BALANCE SET
 1331          BNEQ    80$                     ; NO, CAN OUTSWAP
 1332          CLRL    FP                      ; PREVENT OUTSWAP SCHEDULE
 1333 80$:     BRW     OUTSWAP                 ; TRY TO FORCE AN OUTSWAP
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 28
X-35U3 SCHEDULE SWAP 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (12)

 1336          .SBTTL  SCHEDULE SWAP
 1337
 1338 ;++
 1339 ; FUNCTIONAL DESCRIPTION:
 1340 ;       SWAPSCHED IS CALLED BY THE MAIN LOOP OF THE SWAPPER PROCESS TO CHECK
 1341 ;       ELIGIBLE INSWAP CANDIDATES AND TO PROVIDE MEMORY NEEDED FOR THEIR
 1342 ;       INSWAP.  A QUICK EXIT WILL BE TAKEN IF THE SWAPPER IS ALREADY BUSY.
 1343 ;       NO OUTSWAP WILL BE NEEDED IF THE NUMBER OF REQUIRED PAGES CAN BE
 1344 ;       TAKEN FROM THE FREE PAGE LIST LEAVING AT LEAST FREELIM STILL FREE.
 1345 ;       OTHERWISE OUTSWAP WILL BE ENTERED TO MAKE PAGES AVAILABLE BY ANY
 1346 ;       MEANS NECESSARY.
 1347 ;
 1348 ; ENVIRONMENT:
 1349 ;       ENTERED AT IPL = ASTDEL, NO SPINLOCKS HELD
 1350 ;       EXIT AT IPL = 0.
 1351 ;
 1352 ;--
 1353          .ENABLE LOCAL_BLOCK
 1354 SWAPSCHED:                                ;
 1355          LOCK    LOCKNAME=MMG,-           ; LOCK MMG DATABASE
 1356                  PRESERVE=NO              ; OK to destroy R0
 1357          LOCK    LOCKNAME=SCHED,-         ; LOCK SCHED DATABASE
 1358                  CONDITION=NOSETIPL,-     ; (WELL, DON'T REALLY SET THE IPL)
 1359                  PRESERVE=NO              ; OK to destroy R0
 1360          BBSS    S^#SCH$V_SIP,G^SCH$GB_SIP,5$     ; EXIT IF SWAP IN PROGRESS
 1361          FFS     #0,#32,G^SCH$GL_COMOQS,R2; FIND HIGHEST PRIORITY QUEUE
 1362          BNEQ    10$                      ; FOUND ONE
 1363          BBCC    S^#SCH$V_SIP,G^SCH$GB_SIP,5$     ; CLEAR SWAP IN PROGRESS
 1364 5$:      UNLOCK  LOCKNAME=MMG,-           ; UNLOCK MMG DATABASE, LEAVE IPL
 1365                  PRESERVE=NO              ; OK to destroy R0
 1366          UNLOCK  LOCKNAME=SCHED,-         ; UNLOCK SCHED DATABASE
 1367                  NEWIPL=#0,-              ; LOWER IPL
 1368                  PRESERVE=NO              ; OK to destroy R0
 1369          RSB                              ; AND RETURN
 1370
 1371 10$:     ;
 1372          ; Start doing work
 1373          ;
 1374          ; IPL = SYNCH, MMG AND SCHED SPINLOCKS MUST BE ACQUIRED.
 1375          ;
 1376          PUSHAB  5$                       ; Push return address
 1377                                           ; - a cheap subroutine call
 1378          ;
 1379          ; Perform INSWAP operation
 1380          ;
 1381          PUSHR   #^M<R6,R7,R8,R9,R10,R11,AP,FP>  ; SAVE REGS OTHER THAN R0-R5
 1382          MOVAQ   G^SCH$AQ_COMOH[R2],R3    ; COMPUTE ADDRESS OF QUEUE HEADER
 1383          MOVL    (R3),R4                  ; GET PCB ADDRESS
 1384 ;
 1385 ; THE FOLLOWING CHECK IS NEEDED DUE TO THE ODIOUS MISLEADING SYMPTOMS THAT
 1386 ; MIGHT OTHERWISE RESULT.
 1387 ;
 1388          CMPB    #DYN$C_PCB,PCB$B_TYPE(R4); IS THIS A GOOD PCB?
 1389          BNEQ    QEMPTY                   ;  BUGCHECK IF NOT
 1390 ;
 1391 ; DETERMINE THE SIZE OF THE INSWAP CANDIDATE, TAKING INTO ACCOUNT THE FACT
 1392 ; THAT THE PROCESS HEADER MIGHT ALREADY BE RESIDENT.
```

```
1393 ;
1394         MOVL    PCB$L_PPGCNT(R4),R0     ; COUNT OF PROCESS PAGES
1395         MOVL    PCB$L_GPGCNT(R4),R10    ; COUNT OF GLOBAL PAGES
1396         ADDL    R0,R10                  ; SUM PAGE COUNTS
1397         BBC     #PCB$V_PHDRES,PCB$L_STS(R4),15$ ; CONTINUE IF HEADER NON-RESIDENT
1398         MOVZWL  PCB$W_APTCNT(R4),R0     ; GET ACTIVE PAGE TABLE COUNT
1399         SUBL    R0,R10                  ; SUBTRACT RESIDENT HEADER PAGES FROM REQUIR
1400 15$:                                   ;
1401         SUBL3   G^SCH$GL_FREELIM,G^SCH$GL_FREECNT,R0    ; COMPUTE PAGES AVAILABLE
1402         MOVB    PCB$B_PRI(R4),G^SWP$GB_ISWPRI   ; SAVE PRIORITY OF INSWAP
1403         SUBL3   R10,R0,FP                       ; WILL PROCESS FIT?
1404         BLSS    20$                     ; NO, MUST OUTSWAP
1405         BRW     INSWAP                  ; YES PERFORM SWAP
1406
1407 QEMPTY: BUG_CHECK QUEUEMPTY,FATAL       ; EMPTY QUEUE OR NOT A PCB
1408
1409 ;
1410 ; IF INSWAPPING A NON-REAL TIME PROCESS, THEN CHECK TO SEE IF ITS CURRENT
1411 ; PRIORITY IS THE DEFAULT BACKGROUND PRIORITY.  IF SO, THEN DELAY AT LEAST
1412 ; SWAPRATE INTERVAL SINCE THE LAST INSWAP.  THE EFFECT WILL BE TO AVOID FILLING
1413 ; THE BALANCE SET WITH CRUNCHING PROCESSES IMMEDIATELY.
1414 ;
1415 20$:                                   ; SCHEDULE OUTSWAP
1416         CMPB    #16,PCB$B_PRI(R4)       ; IS THIS A REAL TIME PROCESS?
1417         BGTR    40$                     ; BR IF SO
1418         SUBB3   G^SYS$GB_DEFPRI,#31,R1  ; CONVERT PRIORITY TO INTERNAL FORM
1419         CMPB    R2,R1                   ; IS THIS A CRUNCHER OR LOW PRIORITY JOB?
1420         BLSS    40$                     ; BR IF NOT
1421         MOVL    G^EXE$GQ_SYSTIME,R1     ; GET CURRENT TIME IN APPROX. 10MS UNITS
1422         CMPL    R1,G^SWP$GL_SWTIME      ; HAS INTERVAL ELAPSED?
1423         BGTRU   40$                     ; BR IF YES
1424         BRW     SWAPEXIT                ; CAN'T DO SWAP NOW
1425 40$:                                   ;
1426         .DISABLE  LOCAL_BLOCK
```

```
1429          .SBTTL  OUTSWAP
1430
1431 ;----------------------------------------------------------------------
1432 ;
1433 ;       SCHEDULE AND PERFORM OUTSWAPS IF POSSIBLE
1434 ;
1435 ;----------------------------------------------------------------------
1436 ; FUNCTIONAL DESCRIPTION:
1437 ;       THE OUTSWAP STRATEGY IS TO FREE PROCESS HEADERS FOR OUTSWAP PROCESSES,
1438 ;       USE AVAILABLE MODIFIED PAGES (AFTER WRITING THEM) AND FINALLY AS A LAST
1439 ;       RESORT OUTSWAP ANOTHER PROCESS.  ONLY ONE OF THESE ACTIONS WILL BE TAKEN
1440 ;       AT A TIME THEN THE SCHEDULING SITUATION WILL BE RE-EVALUATED.  THE VALUE
1441 ;       IN FP INDICATES THE SIZE OF THE PAGE DEFICIT AND WILL BE SET POSITIVE IF
1442 ;       ENTERED FROM BALANCE TO FREE DELETED PROCESS HEADERS.
1443 ;
1444 ; INPUT:        FP - NEGATIVE VALUE WILL PERMIT PROCESS OUTSWAP
1445 ;                    ZERO OR POSITIVE WILL PURGE HEADERS ONLY.
1446 ;
1447
1448 OUTSWAP:                                       ; TRY TO OUTSWAP
1449          SUBL3   #1,G^SGN$GL_BALSETCT,R8       ; INIT INDEX FOR BALANCE SLOT SCAN
1450          MCOML   #0,R9                         ; INDICATE NO FREE LIST PURGE CANDIDATE
1451 10$:     TSTW    @PHV$GL_REFCBAS[R8]           ; IS SLOT IN NEED OF CLEANUP?
1452          BNEQ    12$                           ; CONTINUE IF NOT RELEASABLE
1453          BRW     60$                           ; GO RELEASE PAGE TABLES AND HEADER
1454 12$:     CVTWL   @PHV$GL_PIXBAS[R8],R4         ; GET PROCESS INDEX
1455          BLEQ    15$                           ; DELETED PROCESS OR VACANT SLOT
1456          MOVL    @W^SCH$GL_PCBVEC[R4],R4       ; GET PCB ADDRESS FOR PIX
1457          BLBS    PCB$L_STS(R4),20$             ; SKIP IF PROCESS IS RESIDENT
1458 15$:     BEQL    20$                           ; VACANT SLOT
1459          TSTB    G^EXE$GQ_SYSTIME              ; ADD 1 IN 8 RANDOMNESS TO DECISION
1460          BEQL    17$                           ; BRANCH ON LOW PROBABILITY
1461          TSTL    R9                            ; CHECK FOR REMEMBERED INDEX
1462          BGEQ    20$                           ; YES DONT OVERWRITE
1463 17$:     MOVL    R8,R9                         ; SAVE BALANCE SLOT NUMBER OF CANDIDATE
1464 20$:     SOBGEQ  R8,10$                        ; TRY ALL SLOTS
1465          TSTB    G^EXE$GQ_SYSTIME+1            ; ADD 1 IN 256 RANDOMNESS TO DECISION
1466          BEQL    21$                           ; BRANCH ON VERY LOW PROBABILITY
1467          MOVL    R9,R8                         ; GET AND TEST SLOT INDEX FOR SECONDARY CAND
1468          BGEQ    24$                           ; BR IF SLOT FOR CLEANUP
1469 21$:     TSTL    FP                            ; CHECK FOR HEADER PURGE
1470          BGEQ    22$                           ; EXIT IF SO
1471 ;
1472 ; SINCE THERE WAS NO HEADER TO FREE, WE MUST NOW WRITE MODIFIED PAGES OR OUTSWAP
1473 ; SOME PROCESS.  ONLY IF MODIFIED PAGES (MFYCNT-LOLIM) WILL TOTALLY SATISFY OUR
1474 ; NEEDS WILL THEY BE WRITTEN.  OTHERWISE THE LEAST USEFUL (BY SOME OPINION) PROCESS
1475 ; WILL BE OUTSWAPPED AND THE SITUATION RECONSIDERED.
1476 ;
1477          JSB     G^SCH$OSWPSCHED               ; SCHEDULE OUTSWAP
1478          TSTL    R4                            ; ANY CANDIDATE?
1479          BNEQ    23$                           ; YES
1480 22$:     BRW     SWAPEXIT                      ; ELSE EXIT AND TRY LATER
1481 23$:     BRW     70$                           ; GO DO OUTSWAP
1482 ;
1483 ; A HEADER SLOT IN NEED OF CLEANUP WAS FOUND, NOW SCAN THE FREELIST FOR ALL
1484 ; PAGES WHOSE PTE BACK POINTERS PLACE THEM WITHIN THIS HEADER.  DELETE THE
1485 ; CONTENT OF THOSE PAGES VIA G^MMG$DELCONPFN TO FINALLY REDUCE THE REFERENCE
```

```
1486 ; COUNT BINDING THE HEADER.
1487 ;
1488 24$:    ROTL    #2,G^SWP$GL_BSLOTSZ,R7  ; GET SIZE OF BALANCE SLOT IN BYTES
1489         MULL3   R7,R8,R6                ; COMPUTE OFFSET TO BASE OF SLOT
1490         PUSHAB  @SWP$GL_BALSPT[R6]      ; ADD BASE TO GET ADDRESS
1491         ROTL    #7,R6,R2                ; FORM OFFSET TO PHD BASE
1492         PUSHAB  @SWP$GL_BALBASE[R2]     ; BASE ADDRESS FOR PHD
1493         ROTL    #7,R7,R1                ; MUL SPT SLOT SIZE BY 128
1494         PUSHAB  @4(SP)[R7]              ; FORM HIGH LIMIT FOR PAGTBLPPTE
1495         PUSHAB  @4(SP)[R1]              ; ANS SAVE PTE HIGH LIMIT
1496
1497 ;
1498 ;       AT THIS POINT:
1499 ;
1500 ;       00(SP) - HIGH LIMIT ADDRESS FOR PROCESS HEADER
1501 ;       04(SP) - HIGH LIMIT FOR PROCESS PAGE TABLE PTE
1502 ;       08(SP) - LOW LIMIT FOR PROCESS HEADER
1503 ;       12(SP) - LOW LIMIT  FOR PROCESS PAGE TABLE PTE
1504 ;
1505         ASSUME  PFN$C_FREPAGLST EQ 0    ;
1506         TSTW    @PHV$GL_PIXBAS[R8]      ; IS THIS A DELETED PROCESS HEADER?
1507         BGEQ    25$                     ; BR IF NOT.
1508         BUG_CHECK ICPAGELOC,FATAL       ; DELPRC SHOULD HAVE TAKEN CARE OF THIS
1509 25$:    MOVL    G^PFN$AL_TAIL,R0        ; GET TAIL OF FREELIST TO START SCAN
1510         BEQL    45$                     ; NO PAGES, DONE
1511 30$:            PFN_REFERENCE   -
1512         MOVZWL  <@W^PFN$AX_BLINK[R0],R9>,- ; GET BACKWARD LINK
1513                 LONG_OPCODE=MOVL,-
1514                 IMAGE=SYS_NONPAGED
1515         MOVL    @W^PFN$AL_PTE[R0],R3    ; GET SVA OF PTE FOR PAGE
1516         ASSUME  PFN$C_PPGTBL EQ 4
1517         ASSUME  PFN$C_GPGTBL EQ 5
1518         EXTZV   #2,#1,@W^PFN$AB_TYPE[R0],R1     ; GET PAGE TABLE BIT
1519         CMPL    R3,8(SP)[R1]            ; COMPARE WITH LOW LIMIT
1520         BLSSU   40$                     ; OUT OF RANGE
1521         CMPL    R3,(SP)[R1]             ; COMPARE WITH HIGH LIMIT
1522         BGEQU   40$                     ; OUT OF RANGE
1523         CLRL    R2                      ; SET LIST NUMBER FOR DELETE
1524         JSB     G^MMG$REMPFN            ; REMOVE PAGE FROM FREE LIST
1525         BSBW    RELDELPAGE              ; RELEASE PAGE DELETING CONTENT
1526         TSTW    @PHV$GL_REFCBAS[R8]     ; DID WE FREE PROCESS HEADER
1527         BEQL    45$                     ; QUIT NOW
1528 40$:    MOVL    R9,R0                   ; FLINK TO NEXT PAGE
1529         BNEQ    30$                     ; ANOTHER PAGE TO TRY
1530
1531 45$:    TSTW    @PHV$GL_REFCBAS[R8]     ; DID WE FREE PROCESS HEADER
1532         BEQL    50$                     ; YES, RELEASE IT
1533 ;
1534 ; THERE ARE TWO REASONS THAT MIGHT PREVENT THE HEADER FROM BEING RELEASED BY
1535 ; THE FREELIST SCAN:
1536 ;                       1. SOME OF THE TRANSITION PAGES ARE ON THE MODIFED LIST.
1537 ;                       2. THERE IS I/O IN PROGRESS ON THE TRANSITION PAGES.
1538 ; TO COVER THE FORMER CASE (SINCE WE CANT REALLY TELL), THE MODIFIED LIST MUST
1539 ; BE FLUSHED.  HOWEVER THIS IS ACTUALLY QUITE RARE.
1540 ;
1541         MOVL    #MPW$C_SVAPTE,R0        ; Range-based purge request
1542         MOVL    12(SP),R1               ; Low  SVAPTE
```

```
1543          SUBL3   #4,4(SP),R2             ; High SVAPTE (inclusive range)
1544          JSB     G^MMG$PURGEMPL          ; Initiate selective MPL purge
1545          MOVL    #MPW$C_SVAPTE,R0        ; Range-based purge request
1546          MOVL    8(SP),R1                ; Low  SVAPTE
1547          SUBL3   #4,(SP),R2              ; High SVAPTE (inclusive range)
1548          JSB     G^MMG$PURGEMPL          ; Initiate selective MPL purge
1549          ADDL    #16,SP                  ; CLEAN STACK OF LIMITS
1550          MCOML   #0,R9                   ; NO, TRY FOR ANOTHER
1551          BRW     20$                     ; NOW ATTEMPT CLEANUP
1552
1553 50$:     ADDL    #16,SP                  ; CLEAN STACK OF LIMITS
1554 60$:     BRW     RELPHD                  ; GO RELEASE PROCESS HEADER
1555 70$:                                     ;
1556 ;
1557 ;        R4 - PCB OF OUTSWAP CANDIDATE, ALREADY MARKED NON-RESIDENT
1558 ;
1559
1560          MOVL    PCB$L_PHD(R4),R5        ; GET PROCESS HEADER ADDRESS
1561
1562          DECW    G^SWP$GW_BALCNT         ; DECREASE NUMBER IN BALANCE SET
1563          BSBW    OSINIT                  ; INIT REGISTERS FOR SCAN
1564          CLRW    PCB$W_APTCNT(R4)        ; INITIALIZE ACTIVE PAGE TABLE COUNT
1565          MOVL    PHD$L_WSLIST(R5),R7     ; WS INDEX FOR PERM PAGES
1566          MOVL    PHD$L_WSLAST(R5),R6     ; END OF WORKING SET LIST
1567 ;
1568 ;        REGISTER CONVENTIONS FOR OWSLOOP ARE:
1569 ;
1570 ;        R0 - PFN
1571 ;        R1 - SCRATCH, WSLX
1572 ;        R2 - WORKING SET LIST ENTRY (VIRTUAL ADDRESS+FLAGS)
1573 ;        R3 - SVA OF PTE FOR WORKING SET LIST ENTRY
1574 ;        R4 - PCB ADDRESS
1575 ;        R5 - PHD ADDRESS
1576 ;        R6 - END INDEX TO WORKING SET LIST
1577 ;        R7 - WSLX (WORKING SET LIST INDEX)
1578 ;        R8 - PTE CONTENT
1579 ;        R9 - WORKING POINTER TO SWP$AL_MAP
1580 ;        R10 - PTE$M_VALID!PTE$C_ERKW
1581 ;        R11 - BASE ADDRESS OF SWP$AL_MAP
1582 ;
1583 OWSLOOP:                                ; OUTSWAP WS LOOP
1584          MOVL    (R5)[R7],R2             ; GET WORKING SET LIST ENTRY
1585          BLBC    R2,NOTVALID             ; SKIP IF NOT VALID
1586          JSB     G^MMG$SVAPTECHK         ; CONVERT VA TO SVA OF PTE
1587 ;
1588 ;        R3 <- SVA OF PTE FOR VA IN R2
1589 ;
1590          MOVL    (R3),R8                 ; GET CONTENT OF PTE
1591          BLSS    10$                     ; CONTINUE IF VALID PAGE
1592          DECL    R2                      ; CLEAR VALID FLAG
1593 10$:     BICB    #^C<WSL$M_VALID!WSL$M_PAGTYP!WSL$M_PFNLOCK>,R2; ISOLATE INTERESTING
1594          EXTZV   #PTE$V_PFN,#PTE$S_PFN,R8,R0     ; GET PFN FROM PTE
1595          BSBB    OSDISPATCH              ; DISPATCH ON PAGE TYPE
1596 NOTVALID:                               ;
1597          AOBLEQ  R6,R7,OWSLOOP           ; PROCESS ENTIRE WORKING SET LIST
1598          BRB     PROCWRT                 ; DONE WITH WORKING SET LIST, RESET HEADER
1599 OSDISPATCH:                             ;
```

```
1600          ASSUME  WSL$V_VALID    EQ  0
1601          ASSUME  WSL$V_PAGTYP   EQ  1
1602          ASSUME  WSL$V_PFNLOCK  EQ  4
1603          ASSUME  PFN$C_PROCESS  EQ  0
1604          ASSUME  PFN$C_SYSTEM   EQ  1
1605          ASSUME  PFN$C_GLOBAL   EQ  2
1606          ASSUME  PFN$C_GBLWRT   EQ  3
1607          ASSUME  PFN$C_PPGTBL   EQ  4
1608          ASSUME  PFN$C_GPGTBL   EQ  5
1609          MOVAL   (R5)[R7],FP             ; COMPUTE ADDRESS OF WSL ENTRY
1610          CASE    R2,<-                   ; SWITCH ON WSL PAGE TYPE + PTE VALID BIT
1611                  PROCTRANS,-             ; 0 => PROCESS TRANSITION PAGE
1612                  PROCVALID,-             ; 1 => PROCESS VALID PAGE
1613                  WSLERR,-                ; 2 => ????  BUGCHECK
1614                  WSLERR,-                ; 3 => ????  BUGCHECK
1615                  GBLTRANS,-              ; 4 => GLOBAL TRANSITION
1616                  GBLVALID,-              ; 5 => GLOBAL VALID
1617                  GBLWRTTRANS,-           ; 6 => GLOBAL WRITABLE TRANSITION
1618                  GBLWRTVALID-            ; 7 => GLOBAL WRITABLE VALID
1619                  PPGTBLTRANS,-           ; 8 => PROCESS PAGE TABLE TRANSITION
1620                  PPGTBLVALID,-           ; 9 => PROCESS PAGE TABLE VALID
1621                  >,TYPE=B                ;
1622          RSB                             ; SKIP PFN LOCK PAGES
1623
1624  SPACEFAIL:
1625          BUG_CHECK INSSWPFIL,FATAL       ; INSUFFICIENT SWAP FILE SPACE
1626
1627  PROCWRT:                                ; RESET PROCESS HEADER BASE REGISTERS
1628          ASSUME  PCB$L_SWAPSIZE EQ PCB$L_WSSWP+4
1629          MOVAL   PCB$L_WSSWP(R4),R2      ; R2 = address of WSSWP/SWAPSIZE pair
1630          TSTL    (R2)                    ; Swap space allocated ?
1631          BEQL    SPACEFAIL               ; BR if no - fatal error
1632          BGTR    10$                     ; BR if simple, contiguous swap area
1633          ADDL3   #PFLMAP$Q_PTR,(R2),R2   ; R2 = address of 1st pointer in window
1634
1635  10$:    MOVZWL  PCB$W_APTCNT(R4),R0     ; GET COUNT OF ACTIVE PAGE TABLES
1636          MOVL    R4,W^OSWPPCB            ; SAVE ADDRESS OF OUTSWAP PROCESS
1637          SUBL    R11,R9                  ; COMPUTE NUMBER OF PAGES * 4
1638          ROTL    #<32-2>,R9,R4           ; DIVIDE COUNT BY 4
1639          MOVL    R4,W^OSWPPGS            ; SAVE COUNT OF OUTSWAP PAGES
1640          MOVL    R11,R3                  ; SVAPTE FOR OUTSWAP I/O
```

```
1643
1644 ;----------------------------------------------------------------------------

1645 ;
1646 ;       DO OUTSWAP I/O FOR PROCESS HEADER AND BODY
1647 ;
1648 ;----------------------------------------------------------------------------

1649
1650          INCL    G^SWP$GL_OSWPCNT        ; ACCOUNT FOR OUTSWAP
1651          BSBW    SWPWRITE                ; WRITE HEADER AND BODY
1652          BLBS    R0,20$                  ; CONTINUE IF NO I/O ERROR
1653          BUG_CHECK OUTSWPERR,FATAL       ; **** OUT SWAP I/O ERROR
1654 20$:                                     ;
1655
1656          BSBW    RELINIT                 ; INIT REGISTERS FOR RELEASE LOOP
1657          MOVL    PCB$L_PHD(R4),R5        ; GET POINTER TO PHD
1658          MOVZWL  PHD$W_PHVINDEX(R5),R8   ; GET PROCESS HEADER SLOT INDEX
1659 30$:     BICL3   R10,(R9)+,R0            ; GET PAGE NUMBER TO RELEASE
1660          CMPZV   #PFN$V_PAGTYP,#PFN$S_PAGTYP,@W^PFN$AB_TYPE[R0],#PFN$C_GLOBAL    ;
1661          BEQL    80$                     ; PAGE IS GLOBAL, COMPLEX CLEANUP
1662          MOVL    @W^PFN$AL_PTE[R0],R6    ; GET POINTER TO PAGE TABLE FOR  PAGE
1663          BICL    #<PTE$M_VALID!PTE$M_MODIFY>,(R6); CLEAR VALID AND MODIFY
1664          TSTW    @W^PFN$AW_SWPVBN[R0]    ; WAS I/O IN PROGRESS?
1665          BEQL    40$                     ; NO, DONT MARK PAGE MODIFIED
1666          BISB    #PFN$M_MODIFY,@W^PFN$AB_STATE[R0] ; MARK PAGE MODIFIED
1667          CMPZV   #PFN$V_LOC,#PFN$S_LOC,- ; IF THIS WAS READ IN PROGRESS
1668                  @W^PFN$AB_STATE[R0],#PFN$C_RDERR ; AND IS NOW PAGE READ ERROR
1669          BNEQ    40$                     ;
1670          DECREF                          ; AND IF THIS IS THE LAST REFERENCE
1671          BNEQ    60$                     ;
1672          MOVZBL  #PFN$C_BADPAGLST,R2     ; THEN DIVERT THE PAGE TO
1673          JSB     G^MMG$INSPFNT           ; THE BAD PAGE LIST
1674          BRB     60$                     ;
1675 40$:     DECREF                          ; DECREMENT REFERENCE COUNT FOR PAGE
1676          BNEQ    55$                     ; NOT RELEASABLE YET
1677 50$:     JSB     G^MMG$RELPFN            ; RELEASE PFN AS APPROPRIATE
1678          BRB     60$                     ; GO FOR NEXT PAGE
1679 55$:     CMPZV   #PFN$V_LOC,#PFN$S_LOC,@W^PFN$AB_STATE[R0],- ;
1680                  #PFN$C_ACTIVE           ; IS STATE ACTIVE?
1681          BNEQ    60$                     ; NO, THEN LEAVE UNCHANGED
1682          INSV    #PFN$C_RELPEND,-        ; MAKE STATE BE RELEASE PENDING
1683                  #PFN$V_LOC,#PFN$S_LOC,@W^PFN$AB_STATE[R0]; IF SOME I/O OUTSTANDING
1684 60$:     SOBGTR  R7,30$                  ; NEXT PAGE IN LIST
1685          BRW     RELPHD                  ; RELEASE PROCESS HEADER IF POSSIBLE
1686 80$:     DECSHR  GTR=60$,-               ; DECREASE SHARE COUNT FOR PAGE
1687                  IMAGE_FLAG=SYS_NONPAGED
1688          BRB     40$                     ; RELEASE PAGE TO FREE LIST IF REFCNT=0
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 35
X-35U3 RELPHD - RELEASE PROCESS HEADER 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (16)

 1691          .SBTTL  RELPHD - RELEASE PROCESS HEADER
 1692
 1693 ;++
 1694 ; FUNCTIONAL DESCRIPTION:
 1695 ;       RELPHD CHECKS THE REFERENCE  COUNT ON THE PROCESS HEADER
 1696 ;       AND RELEASES THE PAGE TABLES FROM THE PROCESS HEADER WHEN ALL
 1697 ;       OF THEIR PAGES HAVE BEEN RELEASED.  THE PAGE TABLES ARE FIRST WRITTEN
 1698 ;       TO THE SWAP IMAGE IF THEY ARE MARKED AS UPDATED.
 1699 ;
 1700 ; CALLING SEQUENCE:
 1701 ;       BRW/JMP RELPHD
 1702 ;
 1703 ; INPUT PARAMETERS:
 1704 ;       R8 - BALANCE SLOT INDEX FOR HEADER TO BE RELEASED
 1705 ;
 1706 ; OUTPUT PARAMTERS:
 1707 ;       R0-R7,R9,R10 VOLATILE
 1708 ;
 1709 ; SIDE EFFECTS:
 1710 ;       THE PAGE TABLES FROM THE PROCESS HEADER MAY BE WRITTEN TO THE
 1711 ;       SWAP IMAGE FOR THE PROCESS IF THEY HAVE BEEN UPDATED.
 1712 ;
 1713 ;--
 1714
 1715
 1716 RELPHD:                                ;
 1717          TSTW    @PHV$GL_REFCBAS[R8]     ; SEE IF PROCESS HEADER IS RELEASABLE
 1718          BEQL    5$                      ; YES, FREE ACTIVE PAGE TABLES
 1719          BRW     OSWPEXIT                ; NO, TRY LATER
 1720 5$:      MOVL    G^SWP$GL_BSLOTSZ,R7     ; SET ITERATION COUNT TO WHOLE BALANCE SLOT
 1721          MULL3   R7,R8,R1                ; GET LONG WORD OFFSET TO SLOT
 1722          MOVAL   @SWP$GL_BALSPT[R1],R6   ; POINT TO BASE OF THIS SLOT
 1723          BSBW    OSINIT                  ; INIT REGISTERS FOR SCAN
 1724          CVTWL   @PHV$GL_PIXBAS[R8],R4   ; GET INDEX TO PROCESS IN SLOT
 1725          BLSS    12$                     ; BR IF DELETED PROCESS
 1726          MOVL    @W^SCH$GL_PCBVEC[R4],R4 ; AND TRANSLATE TO PCB ADDRESS
 1727          MOVL    PCB$L_PHD(R4),R5        ; GET PROCESS HEADER ADDRESS
 1728          MOVL    R8,PCB$L_PHD(R4)        ; INDICATE NO PHD FOR PROCESS
 1729          SUBL    R5,PHD$L_P0BR(R5)       ; UNBIAS MEMORY MANAGEMENT BASE REGISTERS
 1730          SUBL    R5,PHD$L_P1BR(R5)       ; FOR BOTH P0 AND P1 SPACE
 1731          BBCC    #PCB$V_PHDRES,PCB$L_STS(R4),7$  ; MARK PHD NON-RESIDENT
 1732 7$:      MOVL    PHD$L_WSLX(R5),AP       ; GET POINTER TO WSLX SAVE AREA
 1733          MOVAL   (R5)[AP],AP             ; AND CONVERT TO BYTE ADDRESS
 1734          MOVL    PHD$L_BAK(R5),FP        ; GET POINTER TO BACKING STORE VECTOR
 1735          MOVAL   (R5)[FP],FP             ; AND CONVERT TO BYTE ADDRESS
 1736          CLRW    PHD$W_EMPTPG(R5)        ; CLEAR COUNT OF EMPTY WSL PAGES
 1737          MOVL    G^SGN$GL_PHDPAGCT,R3    ; GET COUNT OF PHD PAGES EXCLUSIVE OF
 1738          ASHL    #2,R3,R3                ;  PAGE TABLES AND MAKE IT BYTE OFFSET
 1739          ADDL    R6,R3                   ; ADDRESS OF FIRST PT IN BALACE SLOT
 1740 10$:     MOVL    (R6)+,(FP)+             ; COPY ENTRY FROM SPT
 1741          BLSS    15$                     ; BR IF VALID
 1742          CMPL    R6,R3                   ; WHERE IS THIS PAGE?
 1743          BGEQU   11$                     ; BR IF NOT EMPTY WSL PAGE
 1744          INCW    PHD$W_EMPTPG(R5)        ; COUNT EMPTY WSL PAGES
 1745 11$:     CLRL    -4(R6)                  ; ZAP INVALID ENTRY TO NO-ACCESS
 1746                  PFN_REFERENCE   -
 1747          CLRW    <(AP)+>,-               ; AND CLEAR WSLX VALUE FOR PAGE
```

```
1748                    LONG_OPCODE=CLRL,-
1749                    IMAGE=SYS_NONPAGED
1750            BRB     20$                     ;
1751 12$:      BRW     DELPHD                  ; FINISH DELETE FOR PROCESS
1752 15$:      EXTZV   #PTE$V_PFN,#PTE$S_PFN,-4(FP),R0 ; GET PFN FOR VALID ENTRY
1753            BEQL    11$                     ; DEMAND ZERO OR NULL PTE
1754            INSV    @W^PFN$AL_BAK[R0],#PTE$V_BAKX,#PTE$S_BAKX,-4(FP); SAVE BACKUP ADDR
1755                    PFN_REFERENCE   -
1756            MOVW    <@W^PFN$AX_WSLX[R0],(AP)+>,-    ; AND WORKING SET LIST INDEX
1757                    LONG_OPCODE=MOVL,-
1758                    IMAGE=SYS_NONPAGED
1759            BISL3   R0,R10,(R9)+            ; SET INTO SWAPPER MAP
1760            BBSS    #PTE$V_VALID,-4(FP),20$ ; MARK PAGE VALID FOR INSWAP PURPOSES
1761 20$:      SOBGTR  R7,10$                  ; SCAN ENTIRE BALANCE SLOT
1762            SUBL    R11,R9                  ; COMPUTE NUMBER OF PAGES * 4
1763            ASSUME  PCB$L_SWAPSIZE EQ PCB$L_WSSWP+4
1764            MOVAL   PCB$L_WSSWP(R4),R2      ; R2 = address of WSSWP/SWAPSIZE pair
1765            TSTL    (R2)                    ; Simple, contiguous swap area ?
1766            BGTR    25$                     ; BR if yes
1767            ADDL3   #PFLMAP$Q_PTR,(R2),R2   ; R2 = address of 1st pointer in window
1768 25$:      MOVL    R4,W^OSWPPCB            ; SAVE PCB ADDRESS FOR SLOT OWNER
1769            ROTL    #<32-2>,R9,R4           ; DIVIDE COUNT BY 4
1770            MOVL    R4,W^OSWPPGS            ; SAVE COUNT OF OUTSWAP PAGES
1771            MOVL    R11,R3                  ; SET SVA OF MAP FOR I/O
1772            INCL    G^SWP$GL_HOSWPCNT       ; ACCOUNT FOR HEADER OUTSWAP
1773            CLRL    R0                      ; Pages to skip in mapping window
1774            BSBW    SWPWRITE                ; WRITE ACTIVE PAGE TABLES
1775            BLBS    R0,30$                  ; CONTINUE IF NO ERROR
1776            BUG_CHECK APTWRTERR,FATAL       ; **** ACTIVE PAGE TABLE SWAP I/O ERROR
1777 30$:      BSBW    RELINIT                 ; INIT REGISTERS FOR RELEASE LOOP
1778            MOVL    PCB$L_PHD(R4),R8        ; RESTORE BALANCE SLOT INDEX
1779 40$:      BICL3   R10,(R9)+,R0            ; ISOLATE PAGE FRAME NUMBER
1780            MOVL    @W^PFN$AL_PTE[R0],R6    ; GET PTE ADDRESS
1781            MOVL    R0,(R6)                 ; MAKE PTE CORRECT BUT INVALID
1782            DECW    @W^PFN$AW_REFCNT[R0]    ; DROP REFERENCE COUNT
1783            BEQL    50$                     ; MUST BE ZERO
1784            BUG_CHECK APTREFHIGH,FATAL      ; INCONSISTENT PAGE TABLE REFERENCE COUNT
1785 50$:      MOVL    R6,R3                   ; SVAPTE FOR DELCON
1786            BSBW    RELDELPAGE              ; RELEASE PAGE THROUGH DELCONPFN
1787            CLRL    (R6)                    ; SET NO ACCESS ON PFN
1788            SOBGTR  R7,40$                  ; CONTINUE FOR ALL ACTIVE PAGE TABLES
1789            MNEGW   #1,@PHV$GL_REFCBAS[R8]  ; MARK BALANCE SLOT AVAIL
1790            CLRW    @PHV$GL_PIXBAS[R8]      ; AND SET PIX TO NULL
1791            CLRL    PCB$L_PHD(R4)           ; AND SEVER CONNECTION WITH PROCESS
1792
1793 OSWPEXIT:                                  ; OUTSWAP COMPLETE
1794            BRW     SWAPRETRY               ; RETRY SWAP SCHEDULE AFTER OUTSWAP
```

```
1797          .SBTTL  DELPHD - DELETE PROCESS HEADER FOR DELETED PROCESS
1798
1799 ;
1800 ; FUNCTIONAL DESCRIPTION:
1801 ;       DELPHD IS ENTERED BY RELPHD IF THE PROCESS INDEX ASSOCIATED WITH
1802 ;       THE BALANCE SLOT IS NEGATIVE INDICATING THE PROCESS HAS BEEN DELETED.
1803 ;       NOW THAT THE REFERENCE COUNT FOR THE HEADER IS ZERO, ALL PAGES AND
1804 ;       BACKING STORE PAGES CAN BE RELEASED PERMITTING RELEASE OF THE BALANCE
1805 ;       SLOT.  AT THIS POINT THE SPT ENTRIES ARE VALID WITH A PFN, DEMAND ZERO,
1806 ;       OR BACKING STORE ADDRESS FORM.  THERE ARE NO REMAINING TRANSITION PAGES.
1807 ;
1808 ; INPUT PARAMETERS:
1809 ;       R1 - PRODUCT OF SGN$C_BSLOTSZ * BALANCE_SLOT_INDEX
1810 ;       R6 - ADDRESS OF FIRST SPT ENTRY FOR THIS BALANCE SLOT
1811 ;       R7 - SGN$C_BSLOTSZ
1812 ;       R8 - BALANCE_SLOT_INDEX
1813 ;       R10- MASK OF PTE$M_VALID!PTE$M_MODIFY!PTE$C_ERKW
1814 ;
1815
1816 DELPHD:                                 ;
1817          ROTL    #9,R1,R5               ; COMPUTE OFFSET TO PHD FROM BASE
1818          ADDL    G^SWP$GL_BALBASE,R5    ; FORM PHD ADDRESS
1819          ASSUME  PHD$S_PRCPGFL EQ 4
1820          PUSHL   PHD$B_PRCPGFL(R5)      ; Save copy of PROCESS/SYSTEM PF mapping
1821          JSB     G^MMG$DASNPRCPGFLS     ; Deassign all process pagefiles
1822 10$:     MOVL    (R6)+,R0               ; GET PTE FROM SPT
1823          BEQL    40$                    ; BR IF EMPTY
1824          BLSS    20$                    ; BR IF VALID
1825          BBS     #PTE$V_TYP1,R0,25$     ; BR IF TYPE 1 (BACKING STORE)
1826 20$:     BICL    R10,R0                 ; ISOLATE PFN
1827          BEQL    30$                    ; SKIP DEMAND ZERO PTE
1828          MOVL    @W^PFN$AL_BAK[R0],R9   ; GET BACKUP ADDRESS
1829          BICB    #<<PTE$M_VALID!PTE$M_MODIFY>@-24>,-1(R6) ; CLEAR VALID AND MODIFY
1830          BSBW    RELDELPAGE             ; RELEASE PAGE
1831          MOVL    R9,R0                  ; GET BACKUP ADDRESS
1832 25$:     BBS     #PTE$V_TYP0,R0,30$     ; BR if not page file backing store
1833          EXTZV   #PTE$V_PRCPGFLX,-      ; Get PROCESS page file index
1834                  #PTE$S_PRCPGFLX,R0,R3
1835          BICL2   #^C<PTE$M_PGFLVBN>,R0  ; Get page file VBN
1836          BEQL    30$                    ; BR if none
1837          CVTBL   (SP)[R3],R3            ; Convert to SYSTEM page file index
1838          BLEQ    50$                    ; Bugcheck if bad index
1839          JSB     G^MMG$DALCPAGFIL       ; Deallocate SYSTEM pagefile page
1840 30$:     CLRL    -4(R6)                 ; ZAP SPT ENTRY
1841 40$:     SOBGTR  R7,10$                 ; RELEASE ENTIRE HEADER
1842          ADDL2   #4,SP                  ; Remove temporary LW
1843          INVALIDATE_TB                  ; INVALIDATE HEADER
1844          MNEGW   #1,@PHV$GL_REFCBAS[R8] ; MARK SLOT EMPTY
1845          CLRW    @PHV$GL_PIXBAS[R8]     ; POINT OWNER PIX AT NULL PROCESS
1846          DECW    G^SCH$GW_DELPHDCT      ; ACCOUNT FOR DELETED HEADER
1847          BRW     SWAPRETRY              ; AND RETRY SWAP ATTEMPT
1848
1849 50$:     BUG_CHECK BADPRCPGFLX,FATAL
1850
```

```
 1852          .SBTTL  GBLTRANS/GBLVALID/GBLWRTVALID - HANDLE GLOBAL PAGES
 1853
 1854 ;
 1855 ;       GBLTRANS HANDLES THE CASE OF A GLOBAL PAGE IN TRANSITION,
 1856 ;       WHICH IMPLIES THAT THE PAGE HAS BEEN FAULTED BUT IS NOT YET
 1857 ;       RESIDENT.  THE WORKING SET LIST ENTRY FOR THIS PAGE WILL BE
 1858 ;       DELETED AND THE PAGE WILL HAVE TO BE FAULTED AGAIN.
 1859 ;
 1860
 1861 GBLTRANS:                                ; TRANSITION GLOBAL PAGE
 1862 GBLWRTTRANS:                             ; TRANSITION WRITABLE GLOBAL PAGE
 1863          INSV    @W^MMG$GL_GPTBASE[R0],#PTE$V_PFN,#PTE$S_PFN,R0  ; GET GLOBAL PFN FRO
 1864
 1865          .ENABL  LSB
 1866 GBLDROP:                                ; DROP GLOBAL PAGE FROM WORKING SET
 1867          MOVL    R7,R1                   ; GET WSL INDEX FOR RELEASE
 1868          PUSHL   R3                      ; SAVE SVAPTE FOR FOLLOWING DECPTREF
 1869          JSB     G^MMG$DELWSLEX          ; DELETE WSL GIVEN INDEX
 1870          POPR    #^M<R3>                 ; RESTORE SVAPTE
 1871          JSB     G^MMG$DECPTREF          ; AND DROP PAGE TABLE REFERENCE
 1872          DECSHR  GTR=10$,-               ; DECREASE SHARE COUNT
 1873                  IMAGE_FLAG=SYS_NONPAGED
 1874 PROCDROP:
 1875          DECREF  GTR=20$                 ; AND REF COUNT IF LAST SHARER
 1876          JSB     G^MMG$RELPFN            ; RELEASE PAGE IF LAST REFERENCE
 1877 10$:                                     ;
 1878          RSB                             ; RETURN FOR NEXT PAGE
 1879 20$:     CMPZV   #PFN$V_LOC,#PFN$S_LOC,@W^PFN$AB_STATE[R0],- ;
 1880                  #PFN$C_ACTIVE           ; CHECK FOR ACTIVE STATE
 1881          BNEQ    30$                     ; NO, THEN LEAVE STATE UNCHANGED
 1882          INSV    #PFN$C_RELPEND,-        ; SET STATE TO RELEASE PENDING IF
 1883                  #PFN$V_LOC,#PFN$S_LOC,@W^PFN$AB_STATE[R0] ; I/O OUTSTANDING
 1884 30$:     RSB                             ;
 1885          .DSABL  LSB
 1886
 1887 ;
 1888 ;       GBLVALID HANDLES A VALID, NON-WRITABLE, PAGE.
 1889 ;
 1890 GBLVALID:                                ; VALID GLOBAL PAGE
 1891 ;
 1892 ;       IF THE GLOBAL PAGE IS BEING ACTIVELY SHARED THEN IT WILL BE DROPPED
 1893 ;       FROM THE WORKING SET AND REFAULTED LATER (PRESUMABLY WITHOUT I/O).
 1894 ;
 1895          BBS     #WSL$V_WSLOCK,(FP),10$  ; DON'T DROP PAGES LOCKED IN WORKING SET
 1896                  PFN_REFERENCE   -
 1897          CMPW    <#1,@W^PFN$AX_SHRCNT[R0]>,-     ; IS THERE ACTIVE SHARING OF THIS PA
 1898                  LONG_OPCODE=CMPL,-
 1899                  IMAGE=SYS_NONPAGED
 1900          BNEQ    GBLWRTVALID             ; YES, DROP IT AND REFAULT LATER
 1901 ;
 1902 ;       OTHERWISE THE PAGE WILL BE WRITTEN TO THE SWAP IMAGE SINCE IT IS
 1903 ;       UNLIKELY TO BE RESIDENT UPON INSWAP.
 1904 ;
 1905 10$:     BISL3   R0,R10,(R9)+            ; SET IN SWAPPER MAP FOR OUT SWAP
 1906          JSB     G^MMG$DECPTREF          ; DROP PAGE TABLE REFERENCE FOR PAGE
 1907 GBLRESET:                                ; RESET SLAVE PTE TO GPTX FORMAT
 1908          SUBL3   G^MMG$GL_GPTBASE,@PFN$AL_PTE[R0],R1     ; GET GPTX FOR PAGE
```

```
1909            ROTL    #<32-2>,R1,R1           ; AND CONVERT TO CORRECT SCALE
1910            ASSUME  PTE$V_TYP0 EQ PTE$S_GPTX
1911            BBSS    #PTE$V_TYP0,R1,10$      ; MARK AS GLOBAL
1912 10$:      BICL3   #<PTE$M_VALID ! -       ; OBTAIN PERMANENT BITS FOR PTE
1913                      PTE$M_TYP0  ! -       ; BY CLEARING ALL OTHERS
1914                      PTE$M_TYP1  ! -       ;
1915                      PTE$M_PFN>,(R3),R2    ; TO FORM TRANSITION GLOBAL PTE
1916            BISL3   R1,R2,(R3)              ; MUST SET ENTIRE PTE AT ONE TIME
1917                                            ; SO THAT I/O CAN SEE CONSISTENT PTE
1918            RSB                             ; RETURN FOR NEXT PAGE
1919
1920 ;
1921 ;       GBLWRTVALID HANDLES THE CASE OF A WRITABLE GLOBAL PAGE.
1922 ;       SUCH PAGES ARE DROPPED FROM THE WORKING SET BEFORE OUTSWAPPING
1923 ;       AND MUST BE SUBSEQUENTLY RE-FAULTED.
1924 ;
1925 GBLWRTVALID:                               ; VALID WRITABLE GLOBAL PAGE
1926            BBCC    #PTE$V_MODIFY,(R3),10$  ; TEST AND CLEAR MODIFY BIT IN SLAVE PTE
1927            BISB    #PFN$M_MODIFY,@W^PFN$AB_STATE[R0] ; AND SAVE MODIFY STATE
1928 10$:      BSBB    GBLRESET                ; RESET PTE
1929            BRW     GBLDROP                 ; DELETE WORKING SET LIST ENTRY
```

```
1932          .SBTTL  PROCTRANS - PROCESS PAGE IN TRANSITION
1933
1934 ;
1935 ;       THIS PAGE IS IN TRANSITION DUE TO THE FACT THAT THE PAGE FAULT
1936 ;       READ OPERATION HAS NOT YET COMPLETED.  IT IS TREATED AS AN
1937 ;       I/O IN PROGRESS PAGE.
1938 ;
1939
1940 PROCTRANS:                                   ; PROCESS PAGE IN TRANSITION
1941          CMPZV   #PFN$V_LOC,#PFN$S_LOC,-     ; IF THIS PAGE COULD NOT
1942                  @W^PFN$AB_STATE[R0],#PFN$C_RDERR ; BE SUCCESSFULLY READ
1943          BNEQ    PROCVALID                   ;
1944          MOVL    R7,R1                       ; DROP IT FROM THE WORKING SET
1945          JSB     G^MMG$DELWSLEX              ; DELETE THE WSL ENTRY GIVEN WSL INDEX
1946          BRW     PROCDROP                    ; AND RELEASE THE PFN IF LAST REF
1947
1948 ;
1949 ;       PROCVALID HANDLES THE CASE OF A VALID PROCESS PAGE WHICH INCURS
1950 ;       SOME SPECIAL PROCESSING IF THERE IS I/O IN PROGRESS.  AN I/O IN
1951 ;       PROGRESS PAGE IS SWAPPED WITH THE BODY OF THE PROCESS TO RESERVE
1952 ;       SPACE FOR IT IN THE SWAP IMAGE AND IS LATER WRITTEN WITH CORRECT
1953 ;       CONTENT BY THE MODIFIED PAGE WRITER TO THIS RESERVED SPACE IN THE
1954 ;       SWAP IMAGE.
1955 ;
1956 PROCVALID:                                  ; PROCESS VALID PAGE
1957          .ENABL  LSB
1958 10$:                                        ;
1959          BBSC    #PFN$V_MODIFY,@W^PFN$AB_STATE[R0],20$   ; BR IF PAGE MODIFIED
1960          BBC     #PTE$V_MODIFY,(R3),30$    ; BR IF PAGE NOT MODIFIED
1961 20$:     BBSS    #WSL$V_MODIFY,(FP),30$    ; SET WORKING SET MODIFIED BIT
1962 30$:                                        ;
1963          CMPW    #1,@W^PFN$AW_REFCNT[R0]   ; CHECK FOR I/O OUTSTANDING
1964          BEQL    40$                       ; NO, NONE
1965          BLBC    R2,SETWRTBAK              ; BRANCH IF TRANSITION PAGE
1966          BBC     #WSL$V_MODIFY,(FP),40$    ; DONT WRITE UNMODIFIED PAGES
1967 SETWRTBAK:                                 ; SET PAGE FOR WRITE BACK TO SWAP FILE
1968          SUBL3   R11,R9,R1                 ; GET OFFSET TO PAGE IN SWAP MAP
1969          DIVL    #4,R1                     ; SCALE BACK TO PAGE NUMBER
1970          MOVW    R1,@W^PFN$AW_SWPVBN[R0]   ; SET OFFSET INTO SWAP IMAGE LESS APTCNT
1971 40$:                                        ;
1972          BISL3   R0,R10,(R9)+              ; PUT PAGE IN SWAPPER MAP
1973 ;
1974 ;       SET DELETE CONTENT FLAG TO CAUSE PAGE TO BE PLACED AT HEAD
1975 ;       OF FREE PAGE LIST AND CONTENT FORGOTTEN.
1976 ;
1977 DELCON: BISB    #PFN$M_DELCON,@W^PFN$AB_STATE[R0] ; SET TO DELETE CONTENT
1978          RSB                               ; RETURN FOR NEXT PAGE
1979          .DSABL  LSB                       ;
1980
1981 WSLERR: BUG_CHECK IVWSETLIST,FATAL         ; INVALID WORKING SET LIST ENTRY
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 41
X-35U3 PAGE TABLE WORKING SET LIST ENTRIES 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (20)

1984          .SBTTL  PAGE TABLE WORKING SET LIST ENTRIES
1985
1986 ;
1987 ;       PAGE TABLE AND PROCESS HEADER ENTRIES IN THE WORKING SET LIST
1988 ;       ARE IGNORED DURING THE PROCESS BODY OUTSWAP SCAN OF THE WORKING
1989 ;       SET LIST.
1990 ;
1991
1992 PPGTBLTRANS:                              ; TRANSITION PAGE TABLE
1993 PPGTBLVALID:                              ; VALID PAGE TABLE
1994          INCW    PCB$W_APTCNT(R4)        ; ACCUMULATE ACTIVE PAGE TABLE COUNT
1995          SUBL    R5,(FP)                 ; UNBIAS WSL VA FOR PAGE TABLE
1996          BBSS    #VA$V_SYSTEM,(FP),10$   ; BUT FORCE SYSTEM BIT ON IN VA
1997 10$:     RSB                             ; RETURN
```

```
2000          .SBTTL  INSWAP
2001
2002 ;-----------------------------------------------------------------------------
--
2003 ;       PERFORM REQUESTED INSWAP
2004 ;
2005 ;       INPUTS:
2006 ;               R4 - PCB ADDRESS OF INSWAP CANDIDATE
2007 ;
2008 ; ENVIRONMENT:
2009 ;       IPL = SYNCH, MMG AND SCHED SPINLOCKS MUST BE HELD.
2010 ;-----------------------------------------------------------------------------
--
2011
2012 INSWAP:                                    ; PERFORM INSWAP
2013          MOVL    PCB$L_PHD(R4),R5          ; GET CURRENT PROCESS HEADER SLOT
2014          BEQL    10$                       ; NONE, MUST ALLOCATE ONE
2015          MOVZWL  PHD$W_PHVINDEX(R5),R8     ; GET BALANCE SLOT INDEX
2016          BRB     40$                       ; AND CONTINUE
2017 10$:     CLRL    R8                        ; INIT INDEX FOR BALANCE SLOT SEARCH
2018 20$:     TSTW    @PHV$GL_REFCBAS[R8]       ; CHECK FOR EMPTY
2019          BLSS    30$                       ; YES, GOT ONE
2020          AOBLSS  G^SGN$GL_BALSETCT,R8,20$; TRY ALL BALANCE SET SLOTS
2021          ROTL    #31,#1,FP                 ; SET FLAG TO PERMIT OUTSWAPPING
2022                                            ; OF PROCESSES
2023          BRW     OUTSWAP                   ; OUTSWAP IF NECESSARY TO GET SLOT
2024 30$:     MOVW    PCB$L_PID(R4),@PHV$GL_PIXBAS[R8]        ; SET PIX FOR BALANCE SET SL
2025          CLRW    @PHV$GL_REFCBAS[R8]       ; AND BUMP REFERENCE COUNT
2026          MULL3   G^SWP$GL_BSLOTSZ,R8,R0    ; COMPUTE BALANCE SLOT OFFSET
2027          ROTL    #9,R0,PCB$L_PHD(R4)       ; MAKE BYTE OFFSET
2028                                            ; POSITIVE UNTIL I/O COMPLETE
2029 40$:     CLRL    R9                        ; INITIALIZE SWAPPER MAP INDEX
2030 50$:     JSB     G^MMG$ALLOCPFN            ; ALLOCATE A PAGE
2031          TSTL    R0                        ; MAKE SURE IT WAS ALLOCATED
2032          BGEQ    60$                       ; YES, CONTINUE
2033          BUG_CHECK INSNFREPAG,FATAL        ; INSUFFICIENT FREE PAGES
2034 60$:     INCW    @W^PFN$AW_REFCNT[R0]      ; REFERENCE PAGE
2035          MOVB    #PFN$C_ACTIVE,@W^PFN$AB_STATE[R0]       ; AND MARK IT ACTIVE
2036          BISL3   #<PTE$C_ERKW!PTE$M_VALID>,R0,@SWP$GL_MAP[R9] ; MARK VALID, WRITABLE
2037          AOBLSS  R10,R9,50$                ; REPEAT FOR ALL REQUIRED PAGES
2038          CLRL    @SWP$GL_MAP[R9] ; PUT STOPPER IN LIST
2039 ;
2040 ;       ALL PAGES HAVE NOW BEEN ACQUIRED AND A BALANCE SET SLOT
2041 ;       ALLOCATED.  THE INSWAP I/O OPERATION CAN NOW BE PERFORMED.
2042 ;
2043          MOVW    G^SCH$GW_SWPFAIL,G^SCH$GW_SWPFCNT       ; RESET FAILURE COUNTER
2044          INCW    G^SWP$GW_BALCNT           ; ADD ONE PROCESS TO BALANCE SET
2045          MOVL    R4,G^SWP$GL_INPCB         ; SAVE POINTER TO IN SWAP PCB
2046          MOVL    R10,G^SWP$GL_ISPAGCNT     ; SAVE COUNT OF ALLOCATED PAGES
2047          MOVW    R8,G^SWP$GW_IBALSETX      ; AND BALANCE SET SLOT NUMBER
2048 ;-----------------------------------------------------------------------
2049 ;
2050 ;       PERFORM INSWAP I/O OPERATION
2051 ;
2052 ;-----------------------------------------------------------------------
2053
2054          ASSUME  PCB$L_SWAPSIZE EQ PCB$L_WSSWP+4
2055          MOVAL   PCB$L_WSSWP(R4),R2        ; R2 = address of WSSWP/SWAPSIZE pair
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 43
X-35U3 INSWAP 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (21)

 2056          TSTL    (R2)
 2057          BEQL    COPYSHELL               ; BRANCH IF SHELL IN SWAP
 2058          BGTR    65$                     ; BR if simple, contiguous swap area
 2059          ADDL3   #PFLMAP$Q_PTR,(R2),R2   ; R2 = address of 1st pointer in window
 2060 65$:     CLRL    R0                      ; Assume full process inswap
 2061          BBC     #PCB$V_PHDRES,PCB$L_STS(R4),70$ ; SWAP EVERYTHING IF HEADER NON-
RES
 2062          MOVZWL  PCB$W_APTCNT(R4),R0     ; GET ACTIVE PAGE TABLE COUNT
 2063 70$:     MOVAL   @SWP$GL_MAP,R3          ; SVA OF PAGE TABLE FOR I/O
 2064          MOVL    R10,R4                  ; NUMBER OF PAGES TO READ
 2065          ADDL2   R10,G^SWP$GL_ISWPPAGES  ; UPDATE TOTAL PAGES INSWAPPED
 2066          INCL    G^SWP$GL_ISWPCNT        ; BUMP INSWAP COUNTER
 2067          BSBW    SWPREAD                 ; PERFORM READ
 2068          BLBS    R0,80$                  ; BRANCH IF NO ERROR IN READ
 2069          BUG_CHECK       INSWAPERR,FATAL ; **** BUGCHECK ON I/O ERROR
 2070 80$:     BRW     SETUP                   ; SET UP PROCESS IN BALANCE SLOT
 2071
 2072 COPYSHELL:
 2073          MOVL    G^MMG$AR_SYSPCB,R4      ; ADDRESS OF SYSTEM PCB
 2074          MOVL    PCB$L_PHD(R4),R5        ; ADDRESS OF SYSTEM PROCESS HEADER
 2075          MOVL    #SWP$C_SHELLSIZ,R6      ; GET I/O PAGE COUNT FOR SHELL
 2076          MOVAB   G^SWP$GL_SHELLBAS,R2    ; GET ADDRESS OF SHELL
 2077          JSB     G^MMG$SVAPTECHK         ; GET ADDRESS OF PAGE TABLE ENTRY
 2078          PUSHR   #^M<R3,R6>              ; SAVE SVAPTE AND PAGE COUNT FOR LATER
 2079          CLRL    R8                      ; SET FLAG INDICATING NO I/O NEEDED
 2080          MOVZWL  #256+3,R7               ; SET FLAGS TO LOCK ONLY VALID OR
 2081                                          ; TRANSITION PAGES AND CREATE OTHERS
 2082                                          ; WITHOUT ZEROING THE PHYSICAL PAGE
 2083          MOVB    #WSL$C_SYSTEM,R2        ; SET PAGE TYPE IN LOW BITS
 2084          BISB    S^#MMG$M_NOWAIT,G^MMG$GB_FREWFLGS ; PREVENT FREWLSE MWAIT
 2085 10$:     JSB     G^MMG$IOLOCKPAG         ; LOCK THE PAGE INTO SYSTEM WORKING SET
 2086          BLBS    R0,20$                  ; BRANCH IF SUCCEEDED
 2087          BUG_CHECK INSNFREPAG,FATAL      ; INSUFFICIENT FREE PAGES
 2088 20$:     BISB    R0,R8                   ; SET FLAG (BIT 1) IF WE HAVE TO I/O IT
 2089          MOVAB   512(R2),R2              ; BUMP VA TO NEXT PAGE
 2090          ADDL    #4,R3                   ; BUMP PTE TO NEXT ENTRY
 2091          SOBGTR  R6,10$                  ; LOOP THROUGH THE PAGES
 2092          POPR    #^M<R3,R4>              ; RECOVER SVAPTE AND PAGE COUNT
 2093          BBC     #1,R8,40$               ; BRANCH IF ALL SHELL PAGES IN MEMORY
 2094          BICB    S^#MMG$M_NOWAIT,G^MMG$GB_FREWFLGS ; ALLOW FREWLSE MWAIT
 2095
 2096                                          ; Mapping window for SHELL I/O
 2097                                          ; is located in SWAPPER's PCB
 2098          FIND_CPU_DATA   R2
 2099          MOVL    CPU$L_CURPCB(R2),R2     ; GET PCB ADDRESS
 2100          ASSUME  PCB$L_SWAPSIZE EQ PCB$L_WSSWP+4
 2101          MOVAL   PCB$L_WSSWP(R2),R2      ; R2 = address of WSSWP/SWAPSIZE pair
 2102          CLRL    R0                      ; Don't skip any pages in window
 2103          BSBW    SWPREAD                 ; PERFORM SHELL READ
 2104          BLBS    R0,30$                  ; BRANCH IF NO ERROR IN READ
 2105          BUG_CHECK INSWAPERR,FATAL       ; **** BUGCHECK ON I/O ERROR
 2106 30$:     BISB    S^#MMG$M_NOWAIT,G^MMG$GB_FREWFLGS ; PREVENT FREWLSE MWAIT
 2107
 2108 40$:     ; ALLOW RESCHEDULES AND PAGEFAULTS WHILE
 2109          ; COPYING SHELL BUT NOT COMPLETION ASTS
 2110
 2111          UNLOCK  LOCKNAME=MMG,-          ; UNLOCK MMG DATABASE, LEAVE IPL
 2112                  PRESERVE=NO             ; OK to destroy R0
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 44
X-35U3 INSWAP 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (21)

 2113          UNLOCK  LOCKNAME=SCHED,-          ; UNLOCK SCHED DATABASE
 2114                  NEWIPL=#IPL$_ASTDEL,-     ; LOWER IPL TO AST
 2115                  PRESERVE=NO               ; OK to destroy R0
 2116          MOVL    #SWP$C_SHELLSIZ,R6        ; GET I/O PAGE COUNT FOR SHELL
 2117          MOVAB   G^SWP$GL_SHELLBAS,R7      ; GET ADDRESS OF SHELL
 2118          ASHL    #9,R6,R0                  ; GET BYTE COUNT
 2119          CLRL    R1                        ; FORM DESTINATION VA
 2120          MOVC3   R0,(R7),(R1)              ; COPY THE SHELL TO LOCATION 0
 2121
 2122          ; BACK TO BLOCKING MMG AND SCHED DATABASES
 2123
 2124          LOCK    LOCKNAME=MMG,-            ; LOCK MMG DATABASE
 2125                  PRESERVE=NO               ; OK to destroy R0
 2126          BICB    S^#MMG$M_NOWAIT,G^MMG$GB_FREWFLGS ; ALLOW FREWLSE MWAIT
 2127          MOVQ    R6,R1                     ; SET UP COUNT AND VA OF SHELL AGAIN
 2128          JSB     G^MMG$SVAPTECHK           ; GET ADDRESS OF PAGE TABLE ENTRY
 2129          JSB     G^MMG$UNLOCK              ; DROP THE REFERENCE COUNTS
 2130                                            ; CONTINUE PROCESS CREATION
 2131          LOCK    LOCKNAME=SCHED,-          ; LOCK SCHED DATABASE
 2132                  CONDITION=NOSETIPL,-      ; (WELL, DON'T REALLY SET IPL)
 2133                  PRESERVE=NO               ; OK to destroy R0
```

```
2136
2137 ;----------------------------------------------------------------------
2138 ;
2139 ;       SET UP PROCESS IN BALANCE SET SLOT
2140 ;
2141 ;----------------------------------------------------------------------
2142
2143 SETUP:                                  ; SETUP INSWAP PROCESS
2144         BSBW    OSINIT                  ; INIT REGISTERS
2145         MOVL    G^SWP$GL_INPCB,R4       ; GET PCB ADDRESS OF INSWAP PROCESS
2146         TSTL    PCB$L_PHD(R4)           ; CHECK FOR NEWLY ALLOCATED PHD
2147         BLSS    10$                     ; BR IF NOT
2148         ADDL    G^SWP$GL_BALBASE,PCB$L_PHD(R4)  ; AND SET ADDRESS IN PCB
2149 10$:    TSTL    PCB$L_WSSWP(R4)         ; CHECK FOR SHELL INSWAP
2150         BNEQ    NOTSHELL                ; BR IF NOT
2151         INVALIDATE_TB                   ; CLEAR TRANSLATION BUFFER
2152         JSB     G^SWP$SHELINIT          ; CALL SHELL INITIALIZATION
2153                                         ; WHICH RETURNS WITH A FULLY INITED PHD
2154 NOTSHELL:                               ;
2155         MOVZWL  G^SWP$GW_IBALSETX,R8    ; AND BALANCE SET INDEX
2156         MULL3   G^SWP$GL_BSLOTSZ,R8,R7  ; COMPUTE OFFSET TO THIS SLOT
2157         MOVAL   @SWP$GL_BALSPT[R7],R7   ; FORM BASE ADDRESS OF MAP FOR SLOT
2158         MOVL    R7,R3                   ; NOW POINT TO PROCESS HEADER
2159         BBSS    #PCB$V_PHDRES,PCB$L_STS(R4),5$  ; SKIP IF PROCESS HEADER STILL RESID
2160         INCL    G^SWP$GL_HISWPCNT       ; COUNT SWAPS INCLUDING HEADER
2161         BSBW    FILLPHD                 ; SET INTO SPT ENTRIES
2162 ;
2163 ;       FILLPHD RETURNS WITH R5 POINTING TO THE PROCESS HEADER POSITION
2164 ;       WITHIN ITS P0 SPACE.
2165 ;
2166         MOVW    R8,PHD$W_PHVINDEX(R5)   ; SET BALANCE SLOT INDEX
2167         ADDL    PCB$L_PHD(R4),PHD$L_P0BR(R5)    ; RELOCATE P0 BASE REGISTER
2168         ADDL    PCB$L_PHD(R4),PHD$L_P1BR(R5)    ; RELOCATE P1 BASE REGISTER
2169         BBSS    #PHD$V_NOACCVIO,PHD$W_FLAGS(R5),5$ ; INDICATE PHD INSWAP TO PAGER
2170 5$:     EXTZV   #0,#PTE$S_PFN,(R7),R0   ; GET PHYSICAL ADDRESS OF PCB
2171         ROTL    #9,R0,R0                ; AND SET IN SOFTWARE PCB
2172         MOVAB   PHD$L_PCB(R0),PCB$L_PHYPCB(R4)  ; ADD OFFSET TO HW PCB
2173 ;
2174 ;       NOW SET PAGES FROM WORKING SET LIST INTO PAGE TABLE ENTRIES
2175 ;
2176         MOVL    PCB$L_PHD(R4),R5        ; GET PROCESS HEADER ADDRESS
2177         INVALIDATE_TB                   ; CLEAR TRANSLATION BUFFER TO SEE IT
2178 ;
2179 ;       A WINDOW IN P1 SPACE IS DOUBLE MAPPED TO ALL OF THE PROCESS
2180 ;       HEADER EXCEPT FOR THE PAGE TABLES. THIS PERMITS REFERENCE TO
2181 ;       MOST OF THE PROCESS HEADER WHILE RUNNING AT IPL LESS THAN THE
2182 ;       SCHEDULER.  TO REFERENCE THE PROCESS HEADER IN SYSTEM SPACE
2183 ;       A PROCESS(OTHER THAN THE SWAPPER) MUST RAISE TO IPL$_SYNCH.
2184 ;
2185
2186         MOVL    G^SWP$GL_PHDBASVA,R2    ; VIRTUAL ADDRESS OF PHD WINDOW
2187         JSB     G^MMG$SVAPTECHK         ; GET POINTER TO WINDOW PTE
2188         MOVL    G^SGN$GL_PHDPAGCT,R2    ; SET COUNT OF PAGES FOR WINDOW
2189         MOVL    #<PTE$C_URKW!PTE$M_VALID>,R1    ; SKELETON PTE
2190 10$:    MOVL    (R7)+,R0                ; GET SWAPPER PTE FOR PHD
2191         BLSS    20$                     ; BR IF VALID PAGE
2192         CLRL    (R3)+                   ; NO, SET NO ACCESS
```

```
2193            BRB     30$                     ;
2194 20$:       INSV    R0,#PTE$V_PFN,#PTE$S_PFN,R1     ; AND INSERT PFN INTO WINDOW PTE
2195            MOVL    R1,(R3)+                ; STORE IN WINDOW AND ADVANCE TO NEX PTE
2196 30$:       SOBGTR  R2,10$                  ; MAP ENTIRE PHD WINDOW
2197 ;
2198 ;         THE REMAINING LIST OF PAGES READ BY THE SWAPPER ARE NOW PROCESSED
2199 ;         ACCORDING TO THE CONTENT OF THE WORKING SET LIST IN THE HEADER OF
2200 ;         THE INSWAP PROCESS.  THE DISPOSITION OF EACH INSWAP PAGE DEPENDS
2201 ;         ON ITS TYPE AND WHETHER THE PAGE IS ALREADY PRESENT IN WHICH CASE
2202 ;         THE NEW, REDUNDANT COPY IS DISCARDED.  SHARED PAGES READ FROM THE
2203 ;         SWAP IMAGE WHICH ARE NOT ALREADY RESIDENT BECOME THE MASTER COPY
2204 ;         AS WELL AS SATISFYING THE REQUIREMENT OF THE INSWAP PROCESS.
2205 ;
2206            ROTL    #PTE$V_VALID,#1,R11     ; FORM VALID MASK
2207            MOVL    PHD$L_WSLIST(R5),R6     ; INDEX TO START OF PERM ENTRIES
2208            MOVL    PHD$L_WSLAST(R5),R7     ; POINTER TO LAST WS ENTRY
2209            .ENABL  LSB                     ;
2210 WSLOOP:    MOVL    (R5)[R6],R2             ; GET A WORKING SET ENTRY
2211            BSBB    10$                     ; AND PROCESS IT
2212            AOBLEQ  R7,R6,WSLOOP            ; SCAN ENTIRE WORKING SET LIST
2213 5$:        BICL3   R10,(R9)+,R0            ; GET AND RELEASE EXCESS PAGES
2214            BEQL    7$                      ; BR IF NO MORE
2215            BSBW    RELPAGE                 ; RELEASE AN EXCESS PAGE
2216            BRB     5$                      ; AND TRY FOR ANOTHER
2217 7$:        BRW     SETASTLVL               ; END OF WORKING SET LIST
2218            ASSUME  WSL$V_VALID EQ 0        ; FOR USE OF BLBS
2219 10$:       BLBS    R2,20$                  ; CHECK FOR VALIDITY, BR IF VALID
2220 15$:       RSB                             ; GET NEXT WSL ENTRY IF NOT VALID
2221 17$:       ADDL    R5,R2                   ; REBIAS VA FOR WSL ENTRY
2222            BISL3   R11,R2,(R5)[R6]         ; AND SET SYSTEM BIT IN VA
2223 18$:       RSB                             ; NEXT WORKING SET LIST ENTRY
2224 20$:                                       ;
2225            BLSS    17$                     ; SKIP PAGE TABLE ENTRIES
2226            JSB     G^MMG$SVAPTECHK         ; GET SVA OF PTE FOR PAGE
2227 ;
2228 ;         R0 - ALL BITS EXCEPT PFN FIELD ARE CLEAR
2229 ;         R2 - WS LIST ENTRY
2230 ;         R3 - SVA OF PTE
2231 ;         R4 - INSWAP PROCESS PCB
2232 ;         R5 - PHD ADDRESS FOR INSWAP PROCESS
2233 ;         R6 - WORKING SET INDEX
2234 ;         R7 - END INDEX TO WORKING SET
2235 ;         R8 - BALANCE SET SLOT INDEX
2236 ;         R9 - ADDRESS OF PHYSICAL PAGE POINTER IN SWP$AL_MAP
2237 ;         R10 - PTE$C_ERKW!PTE$M_VALID!PTE$M_MODIFY
2238 ;         R11 - CONSTANT PFN$M_VALID
2239 ;
2240            BICL3   R10,(R9)+,R0            ; GET PFN FROM MAP
2241            BNEQ    30$                     ; GOT A GOOD PFN
2242            BUG_CHECK ZEROPAGE,FATAL        ; ZERO PAGE TABLE ENTRY FROM SWAP MAP
2243 30$:       MOVL    (R3),AP                 ; GET CONTENT OF PTE
2244            BGEQ    35$                     ; PTE VALID => PFN LOCK, NOT SWAPPED
2245 ;
2246 ; IF THE PAGE IS VALID, IT MUST HAVE BEEN LOCKED IN MEMORY AND WAS IGNORED AT
2247 ; OUTSWAP.  THE REDUNDANT PAGE ALLOCATED FOR THIS WORKING SET LIST ENTRY MUST
2248 ; BE RELEASED AFTER ALL WORKING SET LIST ENTRIES ARE PROCESSED.  NO OTHER
2249 ; ACTION IS NEEDED SINCE THE PTE FOR THE LOCKED PAGE IS ALREADY CORRECT.
```

```
2250 ; THE PFN THAT WOULD HAVE MATCHED THIS ENTRY REALLY BELONGS TO THE NEXT
2251 ; WSL WITHOUT PFNLOCK SET, SO THE MAP POINTER IN R9 MUST BE BACKED UP FOR
2252 ; ANOTHER TRY.
2253 ;
2254         BBC     #WSL$V_PFNLOCK,R2,32$   ; ERROR IF PAGE NOT LOCKED IN MEMORY
2255         TSTL    -(R9)                   ; BACK UP IN SWAPPER MAP
2256         RSB                             ; AND CONTINUE WITH NEXT WSL
2257 32$:    BUG_CHECK       ICPAGELOC,FATAL ; INCONSISTENT PTE/WSL
2258 35$:    BBC     #PTE$V_TYP1,AP,NTYP1    ; BR IF NOT TYPE 1
2259         MOVL    R3,@W^PFN$AL_PTE[R0]    ; NOTE LOCATION OF PTE
2260                 PFN_REFERENCE   -
2261         MOVW    <R6,@W^PFN$AX_WSLX[R0]>,-       ; AND POSITION IN WORKING SET
2262                 LONG_OPCODE=MOVL,-
2263                 IMAGE=SYS_NONPAGED
2264         EXTZV   #PFN$V_BAK0,#PFN$S_BAK0,(R3),R1 ; GET BACKING ADR FROM PTE
2265         BBS     #PTE$V_TYP0,R1,40$      ; BR IF SECTION ADDRESS
2266         EXTZV   #PTE$V_PRCPGFLX,-
2267                 #PTE$S_PRCPGFLX,R1,R2   ; Get PROCESS page file index
2268         MOVZBL  PHD$B_PRCPGFL(R5)[R2],R2; Convert to SYSTEM page file index
2269         INSV    R2,#PFN$V_PGFLX,#PFN$S_PGFLX,R1 ; Form backing store address
2270 40$:    MOVL    R1,@W^PFN$AL_BAK[R0]    ; STORE BACKING ADDRESS
2271         MOVB    #PFN$C_ACTIVE,@W^PFN$AB_STATE[R0]; SET PAGE ACTIVE
2272 RECONNECT:                              ; RECONNECT TO PAGE
2273         BICL3   #^C<PTE$M_PROT!PTE$M_OWN>,(R3),R1       ; RETAIN PERMANENT BITS
2274         MOVAL   (R5)[R6],R2             ; GET ADDRESS OF WORKING SET LIST ENTRY
2275         BBCC    #WSL$V_MODIFY,(R2),50$  ; CHECK FOR MODIFIED AND CLEAR
2276         BISB    #PFN$M_MODIFY,@W^PFN$AB_STATE[R0] ; RECORD MODIFY STATE
2277 50$:    BISL    R11,R1                  ; SET VALID BIT FOR PTE
2278         BISL3   R0,R1,(R3)              ; MERGE BITS WITH PFN  AND STORE IN PGTBL
2279         EXTZV   #VA$V_VPN,#VA$S_VPN,R3,R1       ; GET VPN OF PAGE TABLE
2280         INSV    @W^MMG$GL_SPTBASE[R1],#0,#PTE$S_PFN,R0  ; GET PT PFN
2281                                         ; ASSUMES HIGH ORDER BITS OF R0 ARE CLEAR
2282                 PFN_REFERENCE   -
2283         TSTW    <@W^PFN$AX_SHRCNT[R0]>,-        ; CHECK FOR FIRST ACTIVE PAGE
2284                 LONG_OPCODE=TSTL,-
2285                 IMAGE=SYS_NONPAGED
2286         BNEQ    60$                     ; NO, JUST RAISE SHARE COUNT FOR PT
2287                 PFN_REFERENCE   -
2288         MOVZWL  <@W^PFN$AX_WSLX[R0],R1>,-       ; GET INDEX TO WSL ENTRY FOR PAGE TA
2289                 LONG_OPCODE=MOVL,-
2290                 IMAGE=SYS_NONPAGED
2291         BISL    #WSL$M_WSLOCK,(R5)[R1]  ; AND MARK IT LOCKED IN WORKING SET
2292         INCW    PHD$W_PTCNTACT(R5)      ; COUNT ANOTHER ACTIVE PAGE TABLE
2293         INCW    @PHV$GL_REFCBAS[R8]     ; RAISE REFERENCE COUNT OF BALANCE SLOT
2294 60$:                                    ;
2295                 PFN_REFERENCE   -
2296         INCW    <@W^PFN$AX_SHRCNT[R0]>,-        ; INDICATE ANOTHER ACTIVE PAGE FOR P
2297                 LONG_OPCODE=INCL,-
2298                 IMAGE=SYS_NONPAGED
2299         RSB                             ; RETURN TO GET NEXT WSL ENTRY
2300         .DSABL  LSB                     ;
2301 NTYP1:                                  ; GLOBAL OR TRANSITION
2302         BBS     #PTE$V_TYP0,AP,11$      ; BR IF GLOBAL PAGE
2303         EXTZV   #PTE$V_PFN,#PTE$S_PFN,AP,FP     ; GET OLD PFN IF ANY
2304         BEQL    12$                     ; BR IF ZERO PAGE (BUG CHECK)
2305 ;
2306 ;       RELEASE PFN FOR PAGE ALREADY PRESENT
```

```
2307 ;
2308          BSBW    RELPAGE                 ; RELEASE DUPLICATE PAGE
2309          MOVL    FP,R0                   ; GET SAVED PFN
2310          EXTZV   #PFN$V_LOC,#PFN$S_LOC,@W^PFN$AB_STATE[R0],R2
2311          ASSUME  PFN$C_FREPAGLST EQ 0    ;
2312          ASSUME  PFN$C_MFYPAGLST EQ 1    ;
2313          ASSUME  PFN$C_BADPAGLST EQ 2    ;
2314          ASSUME  PFN$C_RELPEND   EQ 3    ;
2315          ASSUME  PFN$C_RDERR     EQ 4    ;
2316          ASSUME  PFN$C_WRTINPROG EQ 5    ;
2317          ASSUME  PFN$C_RDINPROG  EQ 6    ;
2318          ASSUME  PFN$C_ACTIVE    EQ 7    ;
2319          CASE    R2,<-                   ; DISPATCH ON PAGE LOCATION
2320                  20$,-                   ; 0 => FREE PAGE LIST
2321                  20$,-                   ; 1 => MODIFIED PAGE LIST
2322                  60$,-                   ; 2 => BAD PAGE LIST, PAGE READ/WRITE ERR
2323                  30$,-                   ; 3 => RELEASE PENDING
2324                  10$,-                   ; 4 => PAGE READ ERROR
2325                  30$,-                   ; 5 => WRITE IN PROGRESS
2326                  40$,-                   ; 6 => READ IN PROGRESS
2327                  30$>                    ; 7 => ACTIVE ( I/O NOT YET COMPLETE
2328
2329 10$:     BUG_CHECK ICPAGELOC,FATAL       ; INCONSISTENT PAGE LOCATION
2330
2331 11$:     BRW     GLOBAL                  ; GLOBAL PAGE
2332 12$:     BUG_CHECK ZEROPAGE,FATAL        ; ZERO PFN IN PTE
2333
2334 20$:     PUSHL   R3                      ; SAVE SVAPTE
2335          JSB     G^MMG$REMPFN            ; UNLINK PFN FROM FREE OR MODIFY LIST
2336          POPR    #^M<R3>                 ; RESTORE SVAPTE
2337          MOVAL   (R5)[R6],R2             ; COMPUTE ADDRESS OF WSL ENTRY
2338 30$:     BISL    R11,(R3)                ; SET VALID BIT FOR PTE
2339          ASSUME  PFN$V_LOC EQ 0          ; TO USE BISB INSTEAD OF INSV
2340          BISB    #PFN$C_ACTIVE,@W^PFN$AB_STATE[R0]       ;
2341 40$:     BICB    #<PFN$M_DELCON!-        ; CLEAR DELETE AND
2342                  PFN$M_MODIFY>,-         ; MODIFY
2343                  @W^PFN$AB_STATE[R0]     ; FLAGS
2344 45$:     INCW    @W^PFN$AW_REFCNT[R0]    ; RAISE REFERENCE COUNT
2345          CLRW    @W^PFN$AW_SWPVBN[R0]    ; INDICATE NO ALTERNATE LOCATION
2346          MOVAL   (R5)[R6],R2             ; COMPUTE ADDRESS OF WSL ENTRY
2347          BBCC    #WSL$V_MODIFY,(R2),50$  ; CLEAR MODIFY BIT FOR WSL
2348          BISB    #PFN$M_MODIFY,@W^PFN$AB_STATE[R0] ; RECORD PAGE AS MODIFIED
2349 50$:             PFN_REFERENCE   -
2350          MOVW    <R6,@W^PFN$AX_WSLX[R0]>,-       ; SET WORKING SET LIST INDEX FOR PAG
2351                  LONG_OPCODE=MOVL,-
2352                  IMAGE=SYS_NONPAGED
2353          RSB                             ; AND RETURN FOR NEXT PAGE
2354 ;
2355 ; PAGE IS ON THE BAD PAGE LIST.  IT HAS THE FOLLOWING POSSIBLE STATES
2356 ;       1) BADPAG BIT SET IN G^PFN$AB_TYPE => BUG_CHECK
2357 ;       2) SWPVBN CLEAR => PAGE WRITE ERROR, CORRECT COPY OF MODIFY BIT
2358 ;          IS THE LOGICAL OR OF THE WSLE BIT AND THE PFN BIT
2359 ;       3) SWPVBN SET => PAGE READ ERROR, SET RDERR STATE.
2360 ;
2361 60$:     BBS     #PFN$V_BADPAG,@W^PFN$AB_TYPE[R0],10$ ; ERROR IF BADPAG
2362          PUSHL   R3                      ; SAVE PTE ADDRESS
2363          JSB     G^MMG$REMPFN            ; UNLINK PFN FROM THE BAD PAGE LIST
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 49
X-35U3 INSWAP 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (23)

 2364          POPR    #^M<R3>                 ; RESTORE PTE ADDRESS
 2365          MOVAL   (R5)[R6],R2             ; COMPUTE ADDRESS OF WSL ENTRY
 2366          TSTW    @W^PFN$AW_SWPVBN[R0]    ; IF SWPVBN SET, THEN PAGE READ ERROR
 2367          BNEQ    80$                     ; BRANCH IF PAGE READ ERROR
 2368 ;
 2369 ; PAGE WRITE ERROR
 2370 ;
 2371          ASSUME  PFN$V_MODIFY EQ 7
 2372          TSTB    @W^PFN$AB_STATE[R0]     ; IF PFN MODIFY BIT IS SET
 2373          BGEQ    30$                     ;
 2374          BBSS    #WSL$V_MODIFY,(R2),30$  ; THEN JAM THE WSL ENTRY MODIFY BIT
 2375          BRB     30$                     ; AND CONNECT TO THE PAGE
 2376 ;
 2377 ; PAGE READ ERROR
 2378 ;
 2379 80$:     MOVB    #<PFN$M_DELCON ! PFN$C_RDERR>,- ; SET DELCON
 2380                  @W^PFN$AB_STATE[R0]     ; AND PAGE READ ERROR STATE
 2381          BICW    #<WSL$M_MODIFY>,(R2)    ; CLEAN UP WSLE
 2382          BRB     45$                     ; AND LEAVE PTE IN TRANSITION STATE
 2383 ;
 2384 ;        INSWAP GLOBAL PAGE
 2385 ;
 2386 GLOBAL:                                  ; GLOBAL PAGE INSWAP
 2387          EXTZV   #PTE$V_GPTX,#PTE$S_GPTX,AP,R1   ; GET GLOBAL PAGE TABLE INDEX
 2388          MOVAL   @W^MMG$GL_GPTBASE[R1],R1        ; AND CONVERT TO ADDRESS OF GPTE
 2389          MOVL    (R1),R2                 ; PICK UP GLOBAL MASTER PTE
 2390          BLSS    10$                     ; BR IF VALID
 2391          BBS     #PTE$V_TYP0,R2,5$       ; BR IF GLOBAL SECTION TYPE
 2392          EXTZV   #PTE$V_PFN,#PTE$S_PFN,R2,R0     ; GET PFN OF TRANSITION PAGE
 2393          ASSUME  PFN$C_FREPAGLST EQ 0
 2394          EXTZV   #PFN$V_LOC,#PFN$S_LOC,@W^PFN$AB_STATE[R0],R2    ; TEST FOR FREE PAGE
 2395          BEQL    20$                     ; YES, REFAULT IT
 2396          ASSUME PFN$C_RDINPROG EQ <PFN$C_RDERR + 2>
 2397          CASE    R2, < -                 ; DISPATCH ON READ CASES:
 2398                  55$, -                  ;  READ ERROR
 2399                  4$,  -                  ;  WRONG STATE
 2400                  60$ >, -                ;  READ IN PROGRESS
 2401                  LIMIT=#PFN$C_RDERR
 2402 4$:      BUG_CHECK       ICPAGELOC, FATAL        ; WRONG STATE - CRASH SYSTEM
 2403 5$:      BRW     50$                     ; A BRANCH ASSIST
 2404
 2405 10$:                                    ; INSWAP WITH VALID GLOBAL PAGE
 2406          BSBW    RELPAGE                 ; RELEASE REDUNDANT PAGE
 2407          EXTZV   #PTE$V_PFN,#PTE$S_PFN,R2,R0     ; GET PFN FROM MASTER
 2408          BRB     40$                     ; AND GO SETUP SLAVE PTE
 2409 20$:                                    ; GLOBAL ON FREE LIST
 2410          PUSHL   R0                      ; SAVE MASTER PFN
 2411          BICL3   R10,-4(R9),R0           ; GET REDUNDANT PFN
 2412          BSBW    RELPAGE                 ; AND RELEASE IT (PRESERVING R1-R3)
 2413          POPR    #^M<R0>                 ; RESTORE MASTER PFN
 2414          BISL    R11,(R1)                ; SET PAGE VALID
 2415          PUSHR   #^M<R1,R3>              ; SAVE SVAGPTE, SVAPTE
 2416          JSB     G^MMG$REMPFN            ; REMOVE PFN FROM FREELIST
 2417          POPR    #^M<R1,R3>              ; RESTORE SVAGPTE, SVAPTE
 2418          INSV    #PFN$C_ACTIVE,#PFN$V_LOC,#PFN$S_LOC,@W^PFN$AB_STATE[R0] ;
 2419          INCW    @W^PFN$AW_REFCNT[R0]    ; RAISE REFERENCE COUNT
 2420          BRB     40$                     ;
```

```
2421 30$:     EXTZV   #VA$V_VPN,#VA$S_VPN,R1,R1       ; GET VPN OF PAGE TABLE
2422          MOVL    @W^MMG$GL_SPTBASE[R1],R1        ; GET PAGE TABLE PTE
2423          EXTZV   #PTE$V_PFN,#PTE$S_PFN,R1,R1     ; EXTRACT PFN
2424                  PFN_REFERENCE   -
2425          TSTW    <@W^PFN$AX_SHRCNT[R1]>,-        ; CHECK FOR FIRST REFERENCE TO PTABL
2426                  LONG_OPCODE=TSTL,-
2427                  IMAGE=SYS_NONPAGED
2428          BNEQ    35$                     ; NO
2429          BUG_CHECK GBLPAGSZRO,FATAL      ; GLOBAL PAGE SHARE COUNT ZERO
2430 35$:           PFN_REFERENCE   -
2431          INCW    <@W^PFN$AX_SHRCNT[R1]>,-        ; RAISE GLOBAL PAGE TABLE SHARE COUN
2432                  LONG_OPCODE=INCL,-
2433                  IMAGE=SYS_NONPAGED
2434 40$:           PFN_REFERENCE   -
2435          INCW    <@W^PFN$AX_SHRCNT[R0]>,-        ; RAISE SHARE COUNT FOR GLOBAL PAGE
2436                  LONG_OPCODE=INCL,-
2437                  IMAGE=SYS_NONPAGED
2438          BRW     RECONNECT               ; RECONNECT AND REFERENCE PAGE TABLE
2439 50$:     EXTZV   #PFN$V_BAK0,#PFN$S_BAK0,R2,@W^PFN$AL_BAK[R0] ; SAVE BACKING ADDR
2440          BICL    #^C<PTE$M_PROT!PTE$M_OWN>,R2    ; SAVE PROTECTION AND OWNER FIELDS
2441          BISL    R11,R2                  ; SET PTE VALID
2442          BISL3   R0,R2,(R1)              ; AND STORE WITH PFN IN GPT
2443          MOVL    R1,@W^PFN$AL_PTE[R0]    ; SET SVAGPTE IN PFN DATA BASE
2444          MOVB    #PFN$C_ACTIVE,@W^PFN$AB_STATE[R0] ; SET STATE TO ACTIVE
2445          MOVB    #PFN$C_GLOBAL,@W^PFN$AB_TYPE[R0]  ; AND TYPE TO GLOBAL
2446          BRB     30$                     ; NOW GO SETUP SLAVE PTE
2447
2448 55$:     ; PAGE READ ERROR IN GPTE
2449          ;  THE PFN IN THE GPTE WILL BE DEALLOCATED
2450          ;  THE GPTE WILL BE ALTERED TO USE THE PFN FROM THE INSWAP IMAGE
2451          ;  THE DATA BASE WILL BE ADJUSTED AS APPROPRIATE
2452
2453          BICL3   R10, -4(R9), R2                         ; GET SWAP IMAGE PFN.
2454          MOVL    @W^PFN$AL_BAK[R0], @W^PFN$AL_BAK[R2]    ; COPY BACKING STORE.
2455          MOVB    #PFN$C_ACTIVE, @W^PFN$AB_STATE[R2]      ; SET STATE TO ACTIVE.
2456          MOVB    #PFN$C_GLOBAL, @W^PFN$AB_TYPE[R2]       ; SET TYPE TO GLOBAL.
2457          PFN_REFERENCE -                                 ; COPY SHARE COUNT.
2458          MOVW    < @W^PFN$AX_SHRCNT[R0], @W^PFN$Ax_SHRCNT[R2] >, -
2459                  LONG_OPCODE = MOVL, -
2460                  IMAGE=SYS_NONPAGED
2461          CLRL    @W^PFN$AL_PTE[R0]                       ; SETUP FOR AND RELEASE
2462          BSBW    RELPAGE                                 ; READ ERROR PFN.
2463                                                          ; NOBODY CAN USE IT.
2464          MOVL    R2, R0                                  ; SETUP NEW MASTER PFN.
2465          INSV    R0, #PTE$V_PFN, #PTE$S_PFN, (R1)        ; PLANT PFN IN GPTE.
2466          MOVL    R1, @W^PFN$AL_PTE[R0]                   ; PLANT PTE IN DATABASE.
2467          BISL    R11, (R1)                               ; MAKE PTE VALID.
2468          BRB     40$                                     ; JOIN COMMON CODE.
2469
2470 60$:     BISB    #PFN$M_COLLISION,@W^PFN$AB_TYPE[R0]     ; FLAG COLLISION FOR PAGEREA
2471          UNLOCK  LOCKNAME=MMG,-          ; UNLOCK MMG DATABASE, LEAVE IPL
2472                  PRESERVE=NO             ; OK to destroy R0
2473          PUSHR   #^M<R2,R3,R4,R5>        ; SAVE REGS OVER WAIT
2474          FIND_CPU_DATA   R4
2475          MOVL    CPU$L_CURPCB(R4),R4     ; AND SET PCB ADDRESS
2476          JSB     G^SCH$NEWLVL            ; SET ASTLVL CORRECTLY
2477          MOVAQ   G^SCH$GQ_COLPGWQ,R2     ; GET ADDRESS OF WAIT QUEUE
```

```
2478          PUSHL   #0                      ; NULL KERNEL MODE PSL
2479          JSB     G^SCH$WAITK             ; WAIT WITH NO CALL FRAME
2480
2481          ; LOCK BOTH MMG AND SCHED DATABASES AGAIN !
2482
2483          LOCK    LOCKNAME=MMG,-          ; LOCK MMG DATABASE
2484                  PRESERVE=NO             ; OK to destroy R0
2485          LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
2486                  CONDITION=NOSETIPL,-    ; DON'T SET IPL
2487                  PRESERVE=NO             ; OK to destroy R0
2488          POPR    #^M<R2,R3,R4,R5>        ; RESTORE REGS
2489          BICL3   R10,-4(R9),R0           ; RESTORE CURRENT PFN
2490          BRW     GLOBAL                  ; AND ATTEMPT TO REASSOCIATE PAGE
2491
2492 ;
2493 ; Deallocate swap space in PFLMAP and deallocate structure
2494 ;
2495 PFLMAP_WSSWP:
2496          MOVL    R0,R2                   ; PFLMAP address must be in R2
2497          PUSHAB  B^CLRWSSWP              ; Push return address
2498          JMP     G^MMG$DEALLOCSWPAREA2   ; Deallocate swap space and PFLMAP
2499
2500 ;
2501 ;         SET PROPER AST LEVEL
2502 ;
2503 SETASTLVL:                              ;
2504          MOVAL   PCB$L_ASTQFL(R4),R3     ; GET POINTER TO HEAD OF AST QUEUE
2505          MOVL    (R3),R2                 ; GET POINTER TO FIRST AST CONTROL BLOCK
2506          CMPL    R3,R2                   ; IS LIST EMPTY?
2507          BEQL    20$                     ; YES, DONE
2508          CLRL    R0                      ; ASSUME KERNEL MODE
2509          MOVB    ACB$B_RMOD(R2),R3       ; GET ACTUAL MODE
2510          BLSS    10$                     ; BR IF SPECIAL KERNEL AST
2511          EXTZV   #ACB$V_MODE,#ACB$S_MODE,R3,R0 ; GET ACCESS MODE
2512          BICB3   PCB$B_ASTACT(R4),PCB$B_ASTEN(R4),R1     ; CHECK FOR DELIVERABILITY
2513          BBC     R0,R1,20$               ; BR IF NOT PRESENTLY DELIVERABLE
2514 10$:     MOVB    R0,PHD$B_ASTLVL(R5)     ; SET AST LEVEL FOR PROCESS
2515 20$:     BISL    #<<1@PCB$V_RES>!<1@PCB$V_INQUAN>>,PCB$L_STS(R4) ; MARK PROCESS RESID
2516          MOVL    PCB$L_WSSWP(R4),R0      ; Swap space to deallocate ?
2517          BEQL    NOWSSWP                 ; BR if none
2518          BLSS    PFLMAP_WSSWP            ; BR if PFLMAP address
2519          EXTZV   #24,#8,R0,R3            ; Convert page file index
2520          MOVL    @MMG$GL_PAGSWPVC[R3],R3 ; to page file control block address
2521          BICL2   #<<^XFF>@24>,R0         ; Leave only start VBN to deallocate
2522          BICL3   #<1@31>,PCB$L_SWAPSIZE(R4),R1 ; Size of space to deallocate
2523          JSB     G^MMG$DEALLOCSWPAREA    ; Deallocate swap space
2524 CLRWSSWP:
2525          ASSUME  PCB$L_SWAPSIZE EQ PCB$L_WSSWP+4
2526          CLRQ    PCB$L_WSSWP(R4)         ; Show no swap space held
2527 NOWSSWP:
2528          MOVW    G^SCH$GW_QUAN,PHD$W_QUANT(R5)   ; AND GIVE NEW QUANTUM
2529          MOVZBL  PCB$B_PRI(R4),R0        ; GET CURRENT PRIORITY OF PROCESS
2530          SUBB3   R0,#31,R1               ; COMPUTE EXTERNAL PRIORITY FOR COMPARE
2531          CMPB    R1,G^SYS$GB_DEFPRI      ; IS THIS A CRUNCHER?
2532          BGTR    30$                     ; NO, CONTINUE
2533          MOVL    G^EXE$GQ_SYSTIME,R1     ; GET CURRENT TIME IN APPROX. 10MS UNITS
2534          ADDL3   G^SCH$GL_SWPRATE,R1,G^SWP$GL_SWTIME ; SET NEW CRUNCHER INTERVAL
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 52
X-35U3 INSWAP 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (23)

2535 30$:    JSB     G^SCH$CHSEP              ; CHANGE TO RESIDENT COMPUTE
2536 SWAPRETRY:                               ; RETRY SWAP SCHEDULING
2537         FIND_CPU_DATA   R4
2538         MOVL    CPU$L_CURPCB(R4),R4      ; GET PCB ADDRESS
2539         BBSSI   #PCB$V_WAKEPEN,PCB$L_STS(R4),20$        ; SET TO CANCEL HIBER
2540 20$:
2541         .DSABL  LSB
2542 SWAPEXIT:                                ; EXIT SWAPPER
2543         BBCC    S^#SCH$V_SIP,G^SCH$GB_SIP,10$   ; CLEAR SWAP IN PROGRESS
2544 10$:
2545 SWAPEXITA:                               ; ALTERNATE EXIT, LEAVING SIP SET
2546         POPR    #^M<R6,R7,R8,R9,R10,R11,AP,FP>  ; RESTORE REGISTERS
2547         RSB                              ;
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 53
X-35U3 FILLPHD - FILL SPT ENTRIES TO MAP PHD 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (24

2550          .SBTTL  FILLPHD - FILL SPT ENTRIES TO MAP PHD
2551
2552 ;++
2553 ; FUNCTIONAL DESCRIPTION:
2554 ;       FILLPHD SETS THE PTE ENTRIES FOR THE PROCESS HEADER INTO THE
2555 ;       SPT.
2556 ;
2557 ; CALLING SEQUENCE:
2558 ;       BSB/JSB FILLPHD
2559 ;
2560 ; INPUT PARAMETERS:
2561 ;       R3  -  POINTER TO FIRST SPT ENTRY FOR PHD
2562 ;       R9  -  ADDRESS OF SWAPPER MAP ENTRY TO BE MOVED TO SPT
2563 ;       R10 -  PTE$C_ERKW!PTE$M_VALID!PTE$M_MODIFY
2564 ;
2565 ; OUTPUT PARAMETERS:
2566 ;       R2 - DESTROYED
2567 ;       R5 - ZERO
2568 ;       R6 - DESTROYED
2569 ;       R9 - UPDATED
2570 ;       R11 - DESTROYED
2571 ;       AP - DESTROYED
2572 ;       FP - DESTROYED
2573 ;--
2574
2575 FILLPHD:                                ;
2576
2577          CLRL    R5                     ; SET PHD ADDRESS TO SWAPPER P0 SPACE
2578          INVALIDATE_TB                  ; TO SEE CORRECT PROCESS HEADER IN SWAPPER P
2579          CLRL    R11                    ; INIT HEADER PAGE INDEX
2580          MOVZWL  PHD$W_EMPTPG(R5),R6    ; GET COUNT OF EMPTY PAGES
2581          ASHL    #9,R6,R6               ; CONVERT TO BYTE OFFSET
2582          MOVL    PHD$L_WSLX(R5),AP      ; FORM BASE ADDRESS FOR WSLX
2583          MOVAL   (R5)[AP],AP            ; SAVE VECTOR FOR PHD
2584          MOVL    PHD$L_BAK(R5),FP       ; FORM BASE ADDRESS FOR BACKING STORE ADDRES
2585          MOVAL   (R5)[FP],FP            ; VECTOR
2586          SUBL    R6,AP                  ; ACCOUNT FOR EMPTY PAGES
2587          SUBL    R6,FP                  ; BY SUBTRACTING THEIR SPACE
2588          MOVL    G^SWP$GL_BSLOTSZ,R6    ; SET ITERATION COUNT FOR ENTIRE HEADER
2589 10$:     MOVL    (FP)+,(R3)+            ; SET BACKUP FORM OF PTE IN SPT SLOT
2590          BGEQ    30$                    ; DONE IF NOT VALID
2591          BICL3   R10,(R9)+,R0           ; GET PAGE FROM SWAPPER MAP
2592          MOVAL   -(R3),@W^PFN$AL_PTE[R0] ; SET PTE BACK POINTER
2593          EXTZV   #PTE$V_BAKX,-
2594                  #PTE$S_BAKX,(R3),R1    ; Get VBN & PROCESS page file index
2595          EXTZV   #PTE$V_PRCPGFLX,-
2596                  #PTE$S_PRCPGFLX,(R3),R2 ; Get PROCESS page file index
2597          MOVZBL  PHD$B_PRCPGFL(R5)[R2],R2; Convert to SYSTEM page file index
2598          INSV    R2,#PFN$V_PGFLX,#PFN$S_PGFLX,R1 ; Form backing store address
2599          MOVL    R1,@W^PFN$AL_BAK[R0]   ; SAVE IN PFN DATA BASE
2600                  PFN_REFERENCE   -
2601          MOVW    <(AP)[R11],@W^PFN$AX_WSLX[R0]>,-       ; SAVE WORKING SET LIST INDE
2602                  LONG_OPCODE=MOVL,-
2603                  IMAGE=SYS_NONPAGED
2604          BISL3   R0,R10,(R3)+           ; SET VALID PTE FOR PAGE
2605          MOVB    #<PFN$C_ACTIVE!PFN$M_MODIFY>,@W^PFN$AB_STATE[R0] ; MARK PAGE ACTIVE
2606          MOVB    #PFN$C_PPGTBL,@W^PFN$AB_TYPE[R0]       ; STORE TYPE IN PFN DATA BAS
```

```
 2607 30$:    AOBLSS  R6,R11,10$              ; FILL ENTIRE PROCESS HEADER
 2608         RSB
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 55
X-35U3 RELINIT - INITIALIZE REGISTERS FOR PAGE 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (

 2611          .SBTTL  RELINIT - INITIALIZE REGISTERS FOR PAGE RELEASE LOOP
 2612
 2613 ;++
 2614 ; FUNCTIONAL DESCRIPTION:
 2615 ;       RELINIT SETS UP REGISTERS FOR THE PAGE RELEASE LOOPS FOLLOWING
 2616 ;       OUTSWAP I/O OPERATIONS.
 2617 ;
 2618 ; CALLING SEQUENCE:
 2619 ;       BSB/JSB RELINIT
 2620 ;
 2621 ; INPUT PARAMETERS:
 2622 ;       NONE
 2623 ;
 2624 ; OUTPUT PARAMETERS:
 2625 ;       R0  -  0
 2626 ;       R4  -  OUT SWAP PCB ADDRESS (OSWPPCB)
 2627 ;       R7  -  PAGE COUNT TO RELEASE
 2628 ;       R9  -  BASE ADDRESS FOR SWAPPER MAP (SWP$AL_MAP)
 2629 ;       R10 -  PTE$C_ERKW!PTE$M_VALID!PTE$M_MODIFY
 2630 ;       R11 -  BASE ADDRESS FOR SWAPPER MAP (SWP$AL_MAP)
 2631 ;
 2632 ;--
 2633
 2634 RELINIT:                                ; RELEASE LOOP INITIALIZATION
 2635          MOVL    W^OSWPPCB,R4          ; GET PCB ADDRESS OF OUT SWAP PROCESS
 2636          MOVL    W^OSWPPGS,R7          ; AND PAGE COUNT FOR RELEASE LOOP
 2637 ;       BRB     OSINIT                ; FALL INTO OSINIT
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 56
X-35U3 OSINIT - OUTSWAP SCAN REGISTER INITIALIZ 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1

 2640          .SBTTL  OSINIT - OUTSWAP SCAN REGISTER INITIALIZATION
 2641
 2642 ;++
 2643 ; FUNCTIONAL DESCRIPTION:
 2644 ;       OSINIT SETS UP REGISTERS FOR PAGE TABLE SCANS REQUIRED DURING
 2645 ;       OUTSWAPPING.
 2646 ;
 2647 ; INPUT PARAMETERS:
 2648 ;       NONE
 2649 ;
 2650 ; OUTPUT PARAMETERS:
 2651 ;       R9 - BASE ADDRESS OF SWAPPER MAP (SWP$AL_MAP)
 2652 ;       R10 - PTE$C_ERKW!PTE$M_VALID
 2653 ;       R11 - BASE ADDRESS OF SWAPPER MAP (SWP$AL_MAP)
 2654 ;
 2655 ;--
 2656
 2657 OSINIT:                                       ;
 2658          MOVAL   @SWP$GL_MAP,R9                ; SET BASE OF SWAPPER MAP
 2659          MOVL    R9,R11                        ; AND MAKE REFERENCE COPY
 2660          MOVL    #<PTE$C_ERKW!PTE$M_VALID!PTE$M_MODIFY>,R10 ; MASK TO VALIDATE SWAP P
 2661          RSB                                   ; RETURN
```

```
2664          .SBTTL  RELPAGE - RELEASE DUPLICATE PAGE
2665
2666 ;++
2667 ; FUNCTIONAL DESCRIPTION:
2668 ;       RELPAGE RELEASES A PHYSICAL PAGE WHICH DUPLICATES A PAGE ALREADY
2669 ;       PRESENT FOR THE PROCESS.  THIS SITUATION CAN ARISE DUE TO A PARTIAL
2670 ;       INSWAP OR A GLOBAL PAGE WHICH IS ALREADY PRESENT.
2671 ;
2672 ; CALLING SEQUENCE:
2673 ;       BSB/JSB RELPAGE
2674 ;
2675 ; INPUT PARAMETERS:
2676 ;       R0 - PFN TO RELEASE
2677 ;       R3 - SVA OF PTE (RELDELPAGE ONLY)
2678 ;
2679 ; OUTPUT PARAMETERS:
2680 ;       R1 - PRESERVED (RELPAGE ONLY)
2681 ;       R2 - PRESERVED (RELPAGE ONLY)
2682 ;       R3 - PRESERVED (RELPAGE ONLY)
2683 ;
2684 ;--
2685
2686 RELDELPAGE:                             ; RELEASE PAGE THROUGH DELCONPFN
2687          JSB     G^MMG$DELCONPFN        ; DELETE PAGE CONTENT AND INIT PFN DATA
2688 RELPAGE:                                ; RELEASE PAGE
2689          PUSHR   #^M<R1,R2,R3>          ; PRESERVE REGISTERS
2690          CLRB    @W^PFN$AB_STATE[R0]    ; INIT PFN DATA FOR RELEASE
2691          CLRW    @W^PFN$AW_REFCNT[R0]   ; ZERO REFERENCE COUNT
2692          ASSUME  PFN$C_FREPAGLST EQ 0   ;
2693          CLRL    R2                     ; INDICATE FREELIST
2694          JSB     G^MMG$INSPFNH          ; RELEASE PFN TO HEAD OF FREE LIST
2695          POPR    #^M<R1,R2,R3>          ; RESTORE REGISTERS
2696          RSB                            ; AND RETURN TO CALLER
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 58
X-35U3 SWPREAD/SWPWRITE - SWAPPER I/O ROUTINES 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (

 2699          .SBTTL  SWPREAD/SWPWRITE - SWAPPER I/O ROUTINES
 2700
 2701 ;++
 2702 ; FUNCTIONAL DESCRIPTION:
 2703 ;       SWPREAD AND SWPWRITE PERFORM THE DETAIL WORK REQUIRED TO READ
 2704 ;       OR WRITE A SET OF CONTIGUOUS PAGES IN A WORKING SET SWAP IMAGE.
 2705 ;       THE CALL TO EITHER SWPREAD OR SWPWRITE IS ACTUALLY A CO-ROUTINE
 2706 ;       CALL WHICH RETURNS ONLY AFTER ALL SEGMENTS OF THE I/O OPERATION
 2707 ;       HAVE BEEN PERFORMED.  THIS RETURN IS EFFECTED BY A SPECIAL KERNEL
 2708 ;       AST.
 2709 ;
 2710 ; CALLING SEQUENCE:
 2711 ;       BSB/JSB SWPREAD/SWPWRITE
 2712 ;
 2713 ; INPUT PARAMETERS:
 2714 ;       R0 - Pages to skip in mapping window
 2715 ;               = 0             process header outswap or full process inswap
 2716 ;               = APTCNT        process body outswap or inswap
 2717 ;       R2 - Address of WSSWP/SWAPSIZE mapping window
 2718 ;       R3 - SYSTEM VIRTUAL ADDRESS OF PTE
 2719 ;       R4 - PAGE COUNT
 2720 ;
 2721 ;       00(SP) - RETURN ADDRESS AFTER I/O COMPLETION
 2722 ;       04(SP) - SAVED R6
 2723 ;       08(SP) - SAVED R7
 2724 ;       12(SP) - SAVED R8
 2725 ;       16(SP) - SAVED R9
 2726 ;       20(SP) - SAVED R10
 2727 ;       24(SP) - SAVED R11
 2728 ;       28(SP) - SAVED AP
 2729 ;       32(SP) - SAVED FP
 2730 ;       36(SP) - RETURN TO PREVIOUS THREAD
 2731 ;
 2732 ; IMPLICIT INPUTS:
 2733 ;       PAGE FILE TABLE ENTRIES (PFL) SELECTED BY MAPPING WINDOW
 2734 ;
 2735 ; OUTPUT PARAMETERS:
 2736 ;       R0 - COMPLETION STATUS OF I/O OPERATION
 2737 ;
 2738 ; ENVIRONMENT:
 2739 ;       IPL = SYNCH, MMG AND SCHED SPINLOCKS MUST BE HELD.
 2740 ;--
 2741
 2742          ASSUME  IOEA EQ IOROUTINE+4
 2743          ASSUME  TPGCNT EQ IOEA+4
 2744          ASSUME  PFLMAP EQ TPGCNT+4
 2745          ASSUME  RWSSWP EQ PFLMAP+4
 2746          ASSUME  RSVAPTE EQ RWSSWP+4
 2747          ASSUME  RPGCNT EQ RSVAPTE+4
 2748
 2749          .ENABL  LSB
 2750 SWPREAD:                                ; SWAP READ INITIATION
 2751          PUSHAB  G^EXE$BLDPKTSWPR       ; SET ADDRESS OF BUILD PACKET ROUTINE
 2752          BRB     10$                    ;
 2753 SWPWRITE:                               ; SWAP WRITE INITIATION
 2754          PUSHAB  G^EXE$BLDPKTSWPW       ; SET ADDRESS OF BUILD PACKET ROUTINE
 2755 10$:     MOVAB   W^IOROUTINE,R1         ; ADDRESS OF I/O DATA
```

```
SWAPPER WORKING SET SWAPPER 10-MAY-1989 17:00:40 VAX MACRO V5.0-8 Page 59
X-35U3 SWPREAD/SWPWRITE - SWAPPER I/O ROUTINES 21-MAR-1989 08:44:02 [SYS.SRC]SWAPPER.MAR;1 (

 2756          MOVQ    (SP)+,(R1)+             ; SAVE I/O END ACTION ADDRESS
 2757          MOVL    R4,(R1)+                ; Save remaining page cnt (entire xfer)
 2758          POPR    #^M<R6,R7,R8,R9,R10,R11,AP,FP>; RESTORE REGISTERS OTHER THAN STANDAR
 2759
 2760 11$:     BICL3   #<1@31>,4(R2),R5        ; Get pointer page count
 2761          SUBL2   R5,R0                   ; Subtract from pages remaining to skip
 2762          BLSS    13$                     ; Quit if enough pages skipped
 2763          BBS     #31,4(R2),12$           ; Bugcheck if last pointer
 2764          MOVAQ   8(R2),R2                ; Move to next pointer
 2765          BRB     11$
 2766
 2767 12$:     BUG_CHECK INVPFLMAP,FATAL
 2768
 2769                                          ; R0 contains -(excess page count)
 2770 13$:     MOVL    R2,(R1)+                ; Save PFLMAP window pointer
 2771          MNEGL   R0,R4                   ; Set beginning page count
 2772          ADDL3   R5,(R2),R2              ; Set beginning WSSWP
 2773          SUBL2   R4,R2
 2774          MOVL    R2,(R1)                 ; Save for later processing
 2775
 2776 15$:     EXTZV   #24,#8,R2,R0            ; GET SWAP FILE INDEX
 2777          MOVL    @MMG$GL_PAGSWPVC[R0],R0 ; GET BASE ADDRESS OF PAGE FILE TABLE
 2778          MOVL    B^<TPGCNT-RWSSWP>(R1),R5; Get remaining page count (entire xfer)
 2779          CMPL    R5,R4                   ; Remaining page cnt >= window page cnt?
 2780          BGEQ    17$                     ; BR if yes
 2781          MOVL    R5,R4                   ; Else reduce window page count
 2782 17$:     MOVZWL  #<<1@16>-1>,R5          ; SET I/O SIZE
 2783          CMPL    R4,R5                   ; COMPARE REMAINING PGCNT WITH MAX TRANSFER
 2784          BGTR    20$                     ; USE MAXIMUM TRANSFER
 2785          MOVL    R4,R5                   ; SET TRANSFER TO REMAINING PAGES
 2786 20$:     ADDL3   R5,R2,(R1)+             ; SAVE UPDATED DISK ADDRESS
 2787          MOVAL   (R3)[R5],(R1)+          ; AND UPDATED SAVPTE
 2788
 2789          ; UNLOCK MMG AND SCHED DATABASES
 2790
 2791          PUSHL   R0
 2792          UNLOCK  LOCKNAME=MMG,-          ; UNLOCK MMG DATABASE, LEAVE IPL
 2793                  PRESERVE=NO             ; OK to destroy R0
 2794          UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATABASE
 2795                  NEWIPL=#0,-             ; DROP IPL
 2796                  PRESERVE=NO             ; OK to destroy R0
 2797          POPL    R0
 2798          SUBL3   R5,R4,(R1)              ; SAVE REMAINING PAGE COUNT
 2799          SUBL2   R5,B^<TPGCNT-RPGCNT>(R1); Save remaining page cnt (entire xfer)
 2800          PUSHL   R3                      ; SAVE SVAPTE
 2801          PUSHL   PFL$L_WINDOW(R0)        ; GET WINDOW ADDRESS
 2802          ROTL    #9,R5,-(SP)             ; CONVERT PAGES TO BYTE COUNT
 2803          EXTZV   #0,#24,R2,-(SP)         ; AND ISOLATE BLOCK NUMBER
 2804          ADDL    PFL$L_VBN(R0),(SP)      ; ADD BASE VBN
 2805          JSB     G^SMP$GET_CURPCB        ; Get current PCB address
 2806          MOVL    G^EXE$AR_SYSTEM_PRIMITIVES_DATA,R5      ; IRP IS IN PRIVATE DATA ARE
 2807          ASSUME  IOC_GQ_IRPIQ EQ 0       ; ASSUME IRP LIST IS FIRST DATA CELL
 2808          $REMQHI (R5),R5                 ; GET A PACKET IF POSSIBLE
 2809                                          ; NOTE: R0 is destroyed by $REMQHI macro
 2810          BVC     30$                     ; BR IF ONE AVAILABLE
 2811          JSB     G^EXE$ALLOCIRP          ; ALLOCATE ONE THE LONG WAY
 2812          MOVL    R2,R5                   ; SET PACKET ADDRESS IN PROPER REGISTER
```

```
2813 30$:     MOVAB   B^IODONE,IRP$L_ASTPRM(R5); SET ADDRESS FOR COMPLETION
2814          SUBB3   G^SWP$GB_PRIO,#31,IRP$B_PRI(R5) ; SET PRIORITY FOR TRANSFER
2815          POPR    #^M<R0,R1,R2,R3>        ; RESTORE VBN,BYTECNT,WINDOW,SVAPTE
2816          JSB     @W^IOROUTINE            ; CALL READ OR WRITE ROUTINE
2817
2818          ; We must re-acquire the MMG and SCHED databases because we are
2819          ; about to return to the caller's caller, which happend to have
2820          ; taken out the SPINLOCKS in the first place. Therefore, that code
2821          ; is going to want to release the spinlocks.
2822
2823          LOCK    LOCKNAME=MMG,-          ; LOCK MMG DATABASE
2824                  PRESERVE=NO             ; OK to destroy R0
2825          LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
2826                  CONDITION=NOSETIPL,-    ; (WELL, DON'T REALLY SET IPL)
2827                  PRESERVE=NO             ; OK to destroy R0
2828          RSB                             ; AND RETURN TO ORIGINAL CALLER
2829                                          ; (IE  RETURN TO THE CALLER'S CALLER)
2830
2831 IODONE:                                  ; CONTINUATION CALLED AS KERNEL AST
2832                                          ; IPL = IPL$_ASTDEL
2833          PUSHL   IRP$L_IOST1(R5)         ; SAVE COMPLETION STATUS
2834          MOVL    R5,R0                   ; SET PACKET ADDRESS FOR RELEASE
2835          JSB     G^EXE$DEANONPAGED       ; AND RELEASE IT
2836
2837          ; Take out MMG and SCHED spinlocks again !
2838
2839          LOCK    LOCKNAME=MMG,-          ; LOCK MMG DATABASE
2840                  PRESERVE=NO             ; OK to destroy R0
2841          LOCK    LOCKNAME=SCHED,-        ; LOCK SCHED DATABASE
2842                  CONDITION=NOSETIPL,-    ; (WELL, DON'T REALLY SET IPL)
2843                  PRESERVE=NO             ; OK to destroy R0
2844          POPL    R0                      ; Restore status
2845          BSBB    40$                     ; Perform completion work
2846          UNLOCK  LOCKNAME=MMG,-          ; UNLOCK MMG DATABASE, LEAVE IPL
2847                  PRESERVE=NO             ; OK to destroy R0
2848          UNLOCK  LOCKNAME=SCHED,-        ; UNLOCK SCHED DATABASE
2849                  NEWIPL=#IPL$_ASTDEL,-   ; LOWER IPL
2850                  PRESERVE=NO             ; OK to destroy R0
2851          RSB                             ; RETURN TO CALLER
2852
2853 40$:     BLBC    R0,60$                  ; EXIT IF ERROR
2854          MOVAB   W^RWSSWP,R1             ; GET ADDRESS OF REMAINING TRANSFER PARAMS
2855          MOVQ    (R1),R2                 ; RESTORE WSSWP,SVAPTE TO R2,R3
2856          MOVL    B^<RPGCNT-RWSSWP>(R1),R4 ; AND REMAINING PAGE COUNT
2857          BEQL    50$                     ; BR if no more pages in mapping pointer
2858          BRW     15$                     ; CONTINUE IF MORE PAGES REMAIN
2859
2860 45$:     BUG_CHECK INVPFLMAP,FATAL
2861
2862 50$:     TSTL    B^<TPGCNT-RWSSWP>(R1)   ; Test remaining page cnt. (entire xfer)
2863          BEQL    60$                     ; Exit if no more pages
2864          ADDL3   #8,B^<PFLMAP-RWSSWP>(R1),R2; R2 = address of next pointer
2865          MOVL    R2,B^<PFLMAP-RWSSWP>(R1); Save updated address
2866          BBS     #31,-4(R2),45$          ; Bugcheck if last pointer just used
2867          BICL3   #<1@31>,4(R2),R4        ; Set page count from next pointer
2868          MOVL    (R2),R2                 ; Set WSSWP from next pointer
2869          MOVL    R2,(R1)                 ; and save for processing
```

```
 2870          BRW     15$
 2871
 2872 60$:     PUSHR   #^M<R6,R7,R8,R9,R10,R11,AP,FP>; SAVE NON-STANDARD REGISTERS
 2873          JMP     @W^IOEA                 ; AND CONTINUE SWAP
 2874
 2875          .DSABL  LSB                     ;
 2876          .END
```