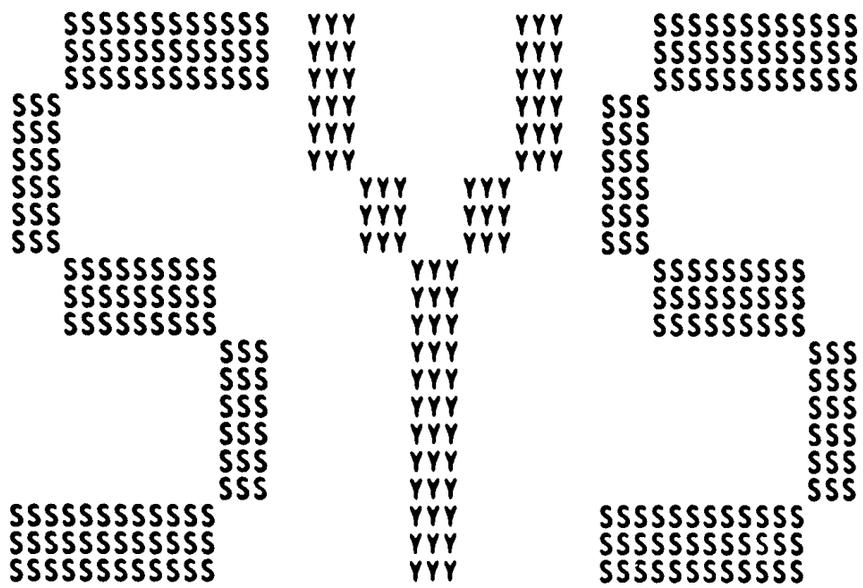

FILEID**POWERFAIL

```
PPPPPPPP      000000    WW      WW  EEEEEEEEEE  RRRRRRRR    FFFFFFFFFF    AAAAAA      IIIIII    LL
PPPPPPPP      000000    WW      WW  EEEEEEEEEE  RRRRRRRR    FFFFFFFFFF    AAAAAA      IIIIII    LL
PP      PP  00      00  WW      WW  EE          RR      RR  FF          AA      AA      II      LL
PP      PP  00      00  WW      WW  EE          RR      RR  FF          AA      AA      II      LL
PP      PP  00      00  WW      WW  EE          RR      RR  FF          AA      AA      II      LL
PP      PP  00      00  WW      WW  EE          RR      RR  FF          AA      AA      II      LL
PPPPPPPP    00      00  WW      WW  EEEEEEEE    RRRRRRRR    FFFFFFF     AA      AA      II      LL
PPPPPPPP    00      00  WW      WW  EEEEEEEE    RRRRRRRR    FFFFFFF     AA      AA      II      LL
PP          00      00  WW  WW  WW  EE          RR  RR      FF          AAAAAAAAAA      II      LL
PP          00      00  WW  WW  WW  EE          RR  RR      FF          AAAAAAAAAA      II      LL
PP          00      00  WWWW  WWWW  EE          RR    RR    FF          AA      AA      II      LL                ....
PP          00      00  WWWW  WWWW  EE          RR    RR    FF          AA      AA      II      LL                ....
PP            000000    WW      WW  EEEEEEEEEE  RR      RR  FF          AA      AA    IIIIII    LLLLLLLLLL        ....
PP            000000    WW      WW  EEEEEEEEEE  RR      RR  FF          AA      AA    IIIIII    LLLLLLLLLL        ....

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

Table of contents

```
0000     1          .TITLE  POWERFAIL - POWER FAIL INTERRUPT HANDLER
0000     2          .IDENT  'V04-000'
0000     3 ;
0000     4 ;****************************************************************************
0000     5 ;*                                                                          *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000     8 ;*  ALL RIGHTS RESERVED.                                                    *
0000     9 ;*                                                                          *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    15 ;*  TRANSFERRED.                                                            *
0000    16 ;*                                                                          *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0000    19 ;*  CORPORATION.                                                            *
0000    20 ;*                                                                          *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    23 ;*                                                                          *
0000    24 ;*                                                                          *
0000    25 ;****************************************************************************
0000    26
0000    27 ;++
0000    28 ;
0000    29 ; Facility:  Executive , Hardware fault handling
0000    30 ;
0000    31 ; Abstract: POWERFAIL contains the code necessary to save the volatile state
0000    32 ;           necessary for restart when power is restored. POWERFAIL also
0000    33 ;           contains the code to restore this state and continue operation
0000    34 ;           upon power restoration.
0000    35 ;
0000    36 ; Environment: MODE=Kernel , IPL=31
0000    37 ;
0000    38 ; Author:  RICHARD I. HUSTVEDT, Creation date:  15-JUN-1978
0000    39 ;
0000    40 ; Modified by:
0000    41 ;
0000    42 ;       V03-016 SRB0125         Steve Beckhardt         06-Jul-1984
0000    43 ;               Clear distributed deadlock detection bitmap expiration
0000    44 ;               timestamps whenever system time is changed to prevent
0000    45 ;               false deadlocks.
0000    46 ;
0000    47 ;       V03-015 WMC0001         Wayne Cardoza           03-May-1984
0000    48 ;               Add support for mount verification of disks.
0000    49 ;
0000    50 ;       V03-014 DWT0208         David W. Thiel          28-Mar-1984
0000    51 ;               Call connection manager on power recovery.
0000    52 ;
0000    53 ;       V03-013 KDM0093         Kathleen D. Morse       6-Feb-1983
0000    54 ;               Added new powerfail codes (16 and 17) for MicroVAX II.
0000    55 ;
0000    56 ;       V03-012 ROW0203         Ralph O. Weber           5-AUG-1983
0000    57 ;               Change EXE$INIT_DEVICE to use the new device driver
```

```
0000     58 ;                  controller and unit initialization routine callers,
0000     59 ;                  IOC$CTRLINIT and IOC$UNITINIT.  These routines provide a
0000     60 ;                  consistant, system-wide interface to the device driver
0000     61 ;                  initialization routines.
0000     62 ;
0000     63 ;          V03-011 TCM0004         Trudy C. Matthews       03-Aug-1983
0000     64 ;                  Add a new error halt bugcheck, defined for the 11/785
0000     65 ;                  processor.
0000     66 ;
0000     67 ;          V03-010 KDM0054         Kathleen D. Morse       11-Jul-1983
0000     68 ;                  Make the cpu-dependent IPR saving be done as close to
0000     69 ;                  the start of the power-down routine as possible for the
0000     70 ;                  Q-bus init.  Change use of PR$_TODR to EXE$READ_TODR.
0000     71 ;                  Move IPR PME into the cpu-dependent save/restore routines.
0000     72 ;
0000     73 ;          V03-009 ROW0188         Ralph O. Weber          30-APR-1983
0000     74 ;                  Fix broken braches to ERL$ routines
0000     75 ;
0000     76 ;          V03-008 TCM0003         Trudy C. Matthews       22-Feb-1983
0000     77 ;                  Add two new error halt bugchecks (defined for 11/790
0000     78 ;                  processors).
0000     79 ;
0000     80 ;          V03-007 KTA3024         Kerbey T. Altmann       31-Dec-1982
0000     81 ;                  Call new routine to do device searching.
0000     82 ;
0000     83 ;          V03-006 TCM0002         Trudy C. Matthews       16-Dec-1982
0000     84 ;                  Initialize R2 before calling CON$SENDCONSCMD.
0000     85 ;
0000     86 ;          V03-005 TCM0001         Trudy C. Matthews       10-Nov-1982
0000     87 ;                  Call CPU-dependent routine CON$SENDCONSCMD to send
0000     88 ;                  "clear warm start" command to the console.  Correct bug
0000     89 ;                  in code that drops IPL to let impending powerfails occur;
0000     90 ;                  if one did occur the saved PC/PSL would wipe out two
0000     91 ;                  registers saved on the stack.  Also, drop IPL to IPL$_POWER-2
0000     92 ;                  instead of IPL$_POWER-1 to allow impending powerfail
0000     93 ;                  interrupts to occur. (Thanks to Ruth Goldenberg.)
0000     94 ;
0000     95 ;          V03-004 KTA3018         Kerbey T. Altmann       03-Nov-1982
0000     96 ;                  Removed adapter initialization to SYSLOA.
0000     97 ;
0000     98 ;          V03-003 KDM0002         Kathleen D. Morse       28-Jun-1982
0000     99 ;                  Added $DCDEF and $DEVDEF.
0000    100 ;
0000    101 ;          V03-002 ROW0093         Ralph O. Weber           4-JUN-1982
0000    102 ;                  In EXE$INIT_DEVICE, correct setup for call to unit
0000    103 ;                  initialization to insure that R3 has primary CSR address
0000    104 ;                  and R4 has secondary CSR address when initialization routine
0000    105 ;                  address is stored in the DDT.
0000    106 ;                  This change is distributed as part of SYS.EXE ECO 15 in
0000    107 ;                  Version 3.1.
0000    108 ;
0000    109 ;
0000    110 ;--
0000    111
0000    112 ;
0000    113 ; Include files:
0000    114 ;
```

```
                     0000   115          $ADPDEF                         ; DEFINE ADAPTER CONTROL BLOCK
                     0000   116          $CONDEF                         ; DEFINE CONSOLE FUNCTION CODES
                     0000   117          $CRBDEF                         ; DEFINE CRB OFFSETS
                     0000   118          $DCDEF                          ; DEFINE ADAPTER TYPES
                     0000   119          $DDBDEF                         ; DEFINE DEVICE DATA BLOCK
                     0000   120          $DDTDEF                         ; DEFINE DRIVER DISPATCH TABLE
                     0000   121          $DEVDEF                         ; DEFINE DEVICE TYPES
                     0000   122          $IDBDEF                         ; DEFINE IDB OFFSETS
                     0000   123          $IPLDEF                         ; DEFINE INTERRUPT PRIORITY LEVELS
                     0000   124          $PRDEF                          ; DEFINE PROCESSOR REGISTER NUMBERS
                     0000   125          $RPBDEF                         ; DEFINE RESTART PARAMETER BLOCK OFFSETS
                     0000   126          $TQEDEF                         ; DEFINE TIMER QUEUE ENTRY OFFSETS
                     0000   127          $UBADEF                         ; DEFINE UBA REGISTERS
                     0000   128          $UCBDEF                         ; DEFINE UNIT CONTROL BLOCK
                     0000   129          $VECDEF                         ; DEFINE VECTOR OFFSETS
                     0000   130
                     0000   131 ;
                     0000   132 ; MACROS:
                     0000   133 ;
                     0000   134
                     0000   135 ;
                     0000   136 ; Equated Symbols:
                     0000   137 ;
00000003             0000   138 RESTRT_POWERUP  =       3               ; Power recovery restart code
00000004             0000   139 RESTRT_IVLISTK  =       4               ; Interrupt stack not valid
00000005             0000   140 RESTRT_DBLERR   =       5               ; Double error restart code
00000006             0000   141 RESTRT_HALT     =       6               ; Halt restart code
00000007             0000   142 RESTRT_ILLVEC   =       7               ; Illegal vector code
00000008             0000   143 RESTRT_NOUSRWCS =       8               ; No user WCS restart code
00000009             0000   144 RESTRT_ERRHALT  =       9               ; Error halt restart code
0000000A             0000   145 RESTRT_CHM      =       10              ; CHMx with IS=1 restart code
                     0000   146 ;
                     0000   147
                 00000000   148          .PSECT  $$$220,LONG             ; Data psect
                     0000   149 EXE$GL_PWRDONE::                         ; End time for power up interval
00000000             0000   150          .LONG   0                       ; Done now
                     0004   151 EXE$GL_PWRINTVL::                        ; Allowable interval in 10MS units
00004650             0004   152          .LONG   100*180                 ; Allow three minutes
                 00000000   153          .PSECT  $AEXENONPAGED,LONG      ; INTERRUPT ROUTINES MUST BE LONGWORD
                     0000   154                                          ; ALIGNED
                     0000   155 ;
                     0000   156 ; Own Storage:
                     0000   157 ;
```

```
                                0000   159          .SBTTL  EXE$POWERFAIL - POWER FAIL INTERRUPT SERVICE ROUTINE
                                0000   160 ;++
                                0000   161 ;
                                0000   162 ; Functional Description:
                                0000   163 ;       EXE$POWERFAIL is entered with IPL=31 as a result of a power fail
                                0000   164 ;       interrupt.  The objective is to save the critical volatile machine
                                0000   165 ;       state as quickly as possible and halt the machine.
                                0000   166 ;
                                0000   167 ; Calling Sequence:
                                0000   168 ;       Powerfail interrupt through Vector at offset 12 in the SCB.
                                0000   169 ;
                                0000   170 ; Input Parameters:
                                0000   171 ;       00(SP) - PC at time of powerfail interrupt
                                0000   172 ;       04(SP) - PSL at tiem of powerfail interrupt
                                0000   173 ;
                                0000   174 ; Implicit Inputs:
                                0000   175 ;       All registers and processor registers.
                                0000   176 ;       Restart Parameter Block located via EXE$GL_RPB
                                0000   177 ;
                                0000   178 ;--
                                0000   179          .LIST   MEB                     ; Show macro expansions
                                0000   180
                                0000   181          .ALIGN  LONG                    ; Exception and Interrupt routines must
                                0000   182                                          ; be longword aligned
                                0000   183 EXE$POWERFAIL::                          ;
                    0000'CF     D5 0000   184          TSTL    W^EXE$GL_PFAILTIM               ; Have we restarted yet?
                         53     12 0004   185          BNEQ    10$                     ; No, wait for restart
                    3FFF 8F     BB 0006   186          PUSHR   #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,AP,FP> ; Save all register
                00000000'EF     16 000A   187          JSB     EXE$REGSAVE             ; Save CPU-specific IPR's
                55  0000'CF     D0 0010   188          MOVL    W^EXE$GL_RPB,R5         ; Get address of restart parameter block
                00A8 C5     10  DB 0015   189          MFPR    #PR$_PCBB,RPB$L_PCBB(R5); Save physical address of current pcb
                00B0 C5     11  DB 001A   190          MFPR    #PR$_SCBB,RPB$L_SCBB(R5); Save physical address of System Control Bl
                00AC C5     0C  DB 001F   191          MFPR    #PR$_SBR,RPB$L_SBR(R5)  ; Save physical address of System page table
                00B4 C5     15  DB 0024   192          MFPR    #PR$_SISR,RPB$L_SISR(R5); Save software interrupt summary register
                00B8 C5     0D  DB 0029   193          MFPR    #PR$_SLR,RPB$L_SLR(R5)  ; Save SPT length
                 00000000'EF    16 002E   194          JSB     EXE$READ_TODR           ; Read time-of-day processor register
                0000'CF     50  D0 0034   195          MOVL    R0,W^EXE$GL_PFAILTIM    ; Save time of day at power fail
                                0039   196 ;
                                0039   197 ; Save all other volatile processor registers on the current stack (ISP)
                                0039   198 ;
                        7E  00  DB 0039   199          MFPR    #PR$_KSP,-(SP)          ; Save kernel stack pointer
                        7E  01  DB 003C   200          MFPR    #PR$_ESP,-(SP)          ; Save exec stack pointer
                        7E  02  DB 003F   201          MFPR    #PR$_SSP,-(SP)          ; Save supervisor stack pointer
                        7E  03  DB 0042   202          MFPR    #PR$_USP,-(SP)          ; Save user stack pointer
                        7E  13  DB 0045   203          MFPR    #PR$_ASTLVL,-(SP)       ; Save AST level
                        7E  08  DB 0048   204          MFPR    #PR$_P0BR,-(SP)         ; Save P0 base register
                        7E  09  DB 004B   205          MFPR    #PR$_P0LR,-(SP)         ; Save P0 length register
                        7E  0A  DB 004E   206          MFPR    #PR$_P1BR,-(SP)         ; Save P1 base register
                        7E  0B  DB 0051   207          MFPR    #PR$_P1LR,-(SP)         ; Save P1 length register
                                0054   208 ;
                                0054   209 ; All volatile machine state necessary for restart has now been saved.
                                0054   210 ; At this point the interrupt stack contains:
                                0054   211 ;
                                0054   212 ;       +-------------------------+
                                0054   213 ;       |          P1LR           | 0-24(SP)
                                0054   214 ;       +-------------------------+
                                0054   215 ;       |          P1BR           |
```

```
                        0054   216 ;       +-------------------------+
                        0054   217 ;       |          POLR           |
                        0054   218 ;       +-------------------------+
                        0054   219 ;       |          POBR           |
                        0054   220 ;       +-------------------------+
                        0054   221 ;       |          ASTLVL         |
                        0054   222 ;       +-------------------------+
                        0054   223 ;       |          USP            |
                        0054   224 ;       +-------------------------+
                        0054   225 ;       |          SSP            |
                        0054   226 ;       +-------------------------+
                        0054   227 ;       |          ESP            |
                        0054   228 ;       +-------------------------+
                        0054   229 ;       |          KSP            |
                        0054   230 ;       +-------------------------+
                        0054   231 ;       |    CPU-specific IPR's   | 28-n(SP)
                        0054   232 ;       +-------------------------+
                        0054   233 ;       |          R0             |
                        0054   234 ;       +-------------------------+
                        0054   235 ;       |          R1             |
                        0054   236 ;       +-------------------------+
                        0054   237 ;       |          R2             |
                        0054   238 ;       +-------------------------+
                        0054   239 ;       |          R3             |
                        0054   240 ;       +-------------------------+
                        0054   241 ;       |          R4             |
                        0054   242 ;       +-------------------------+
                        0054   243 ;       |          R5             |
                        0054   244 ;       +-------------------------+
                        0054   245 ;       |          R6             |
                        0054   246 ;       +-------------------------+
                        0054   247 ;       |          R7             |
                        0054   248 ;       +-------------------------+
                        0054   249 ;       |          R8             |
                        0054   250 ;       +-------------------------+
                        0054   251 ;       |          R9             |
                        0054   252 ;       +-------------------------+
                        0054   253 ;       |          R10            |
                        0054   254 ;       +-------------------------+
                        0054   255 ;       |          R11            |
                        0054   256 ;       +-------------------------+
                        0054   257 ;       |          AP             |
                        0054   258 ;       +-------------------------+
                        0054   259 ;       |          FP             |
                        0054   260 ;       +-------------------------+
                        0054   261 ;       |          PC             |
                        0054   262 ;       +-------------------------+
                        0054   263 ;       |          PSL            |
                        0054   264 ;       +-------------------------+
                        0054   265 ;
00A4 C5   5E   D0       0054   266         MOVL    SP,RPB$L_ISP(R5)        ; Save final interrupt stack pointer
          FE   11       0059   267 10$:    BRB     10$                     ; Wait for power off halt
                        005B   268                                         ; This loop is to avoid halting
                        005B   269                                         ; and confusing the console
                        005B   270                                         ; by inadvertently triggering an
                        005B   271                                         ; automatic restart too soon.
                        005B   272
```

```
                                005B   274          .SBTTL  EXE$RESTART - Restore state and restart after power on
                                005B   275 ;++
                                005B   276 ; Functional Description:
                                005B   277 ;       EXE$RESTART is given control by the restart ROM bootstrap if it
                                005B   278 ;       is determined that memory content is valid, the checksum of the
                                005B   279 ;       restart routine verifies and the restart flag in the Restart Control
                                005B   280 ;       Block is enabled.  Initial entry to EXE$RESTART is made with memory
                                005B   281 ;       management disabled IPL=31 with the stack pointer set to the high
                                005B   282 ;       end of the page containing the restart control block.
                                005B   283 ;
                                005B   284 ; Calling Sequence:
                                005B   285 ;       JMP     @RPB$L_RESTART-^X200(SP)
                                005B   286 ;
                                005B   287 ; Input Parameters:
                                005B   288 ;       SP - Address of RPB+^x200
                                005B   289 ;
                                005B   290 ;--
                                005B   291
                            00000000   292          .PSECT  $AAEXENONPAGED,PAGE     ; Must be in page aligned psect
                                0000   293 EXE$RESTART::                            ; Restart entry point
          55    FE00 CE     9E  0000   294          MOVAB   -512(SP),R5             ; Compute base of RPB
          54    00AC C5     D0  0005   295          MOVL    RPB$L_SBR(R5),R4        ; Get base of SPT
                0C    54    DA  000A   296          MTPR    R4,#PR$_SBR             ; Set SPT base register
          0D    00B8 C5     DA  000D   297          MTPR    RPB$L_SLR(R5),#PR$_SLR  ;  and length register
          11    00B0 C5     DA  0012   298          MTPR    RPB$L_SCBB(R5),#PR$_SCBB; Restore pointer to System Control Block
            53    50  A5     D0  0017   299          MOVL    RPB$L_SVASPT(R5),R3     ; Get virtual address of SPT
      51    FFC00000'8F     D0  001B   300          MOVL    #<<EXE$RESTART-^X80000000>@-9>,R1; VPN of EXE$RESTART
            50    DB  AF     9E  0022   301          MOVAB   EXE$RESTART,R0          ; Physical address of EXE$RESTART
       50   50    F7  8F     78  0026   302          ASHL    #-9,R0,R0               ; Convert to physical page number
               51    50     C2  002B   303          SUBL    R0,R1                   ; Compute delta VPN-PFN
             53    6341     DE  002E   304          MOVAL   (R3)[R1],R3             ; Now compute base address for P0PT
                   50       D6  0032   305          INCL    R0                      ; Get PFN+1 of EXE$RESTART for P0LR
             09    50       DA  0034   306          MTPR    R0,#PR$_P0LR            ; Set dummy P0 length
             08    53       DA  0037   307          MTPR    R3,#PR$_P0BR            ; Set base for P0 page table
          56    00A4 C5     D0  003A   308          MOVL    RPB$L_ISP(R5),R6        ; Get Saved interrupt stack pointer
                            003F   309          INVALID                         ; Clear translation buffer
                                                        MTPR    #0,S^#PR$_TBIA
             39    00       DA  003F
             38    01       DA  0042   310          MTPR    #1,#PR$_MAPEN           ; Enable memory management
          0000004B'9F       17  0045   311          JMP     @#10$                   ; Set PC to system space
             5E    56       D0  004B   312 10$:     MOVL    R6,SP                   ; Now restore correct Stack pointer
             03    5C       D1  004E   313          CMPL    AP,#RESTRT_POWERUP      ; Is this a power recovery?
                   68       13  0051   314          BEQL    POWERUP                 ; Yes
5E  00000000'EF  00000200 8F   C1  0053   315          ADDL3   #512,EXE$GL_RPB,SP      ; Use end of restart page as stack
                            005F   316          CASE    AP,<-                   ; Else switch on restart code
                            005F   317                  20$,-                   ; 4 => Interrupt stack not valid
                            005F   318                  30$,-                   ; 5 => CPU double error halt
                            005F   319                  40$,-                   ; 6 => Halt instruction
                            005F   320                  50$,-                   ; 7 => Illegal I/E vector
                            005F   321                  60$,-                   ; 8 => No user WCS
                            005F   322                  70$,-                   ; 9 => Error pending on Halt
                            005F   323                  80$,-                   ;10 => CHM on ISTK halt
                            005F   324                  90$,-                   ;11 => CHM vector <1:0> .NE. 0
                            005F   325                  100$,-                  ;12 => SCB physical read error
                            005F   326                  110$,-                  ;13 => WCS error correction failed
                            005F   327                  120$,-                  ;14 => CPU ceased execution
                            005F   328                  130$,-                  ;15 => Processor clocks out of synch
                            005F   329                  140$,-                  ;16 = > ACV or TNV during mchk exception
```

```
                                                D 8
                     005F   330                     150$,-                  ;17 = > ACV or TNV during kstk not valid
                     005F   331                     >,LIMIT=#RESTRT_IVLISTK ;
          0D' 04 5C  AF  005F                CASEW   AP,#RESTRT_IVLISTK,S^#<<30001$-30000$>/2>-1
                     0063              30000$:
               0020' 0063                   .SIGNED_WORD    20$-30000$
               0024' 0065                   .SIGNED_WORD    30$-30000$
               0028' 0067                   .SIGNED_WORD    40$-30000$
               002C' 0069                   .SIGNED_WORD    50$-30000$
               0030' 006B                   .SIGNED_WORD    60$-30000$
               0034' 006D                   .SIGNED_WORD    70$-30000$
               0038' 006F                   .SIGNED_WORD    80$-30000$
               003C' 0071                   .SIGNED_WORD    90$-30000$
               0040' 0073                   .SIGNED_WORD    100$-30000$
               0044' 0075                   .SIGNED_WORD    110$-30000$
               0048' 0077                   .SIGNED_WORD    120$-30000$
               004C' 0079                   .SIGNED_WORD    130$-30000$
               0050' 007B                   .SIGNED_WORD    140$-30000$
               0054' 007D                   .SIGNED_WORD    150$-30000$
                     007F              30001$:
                     007F   332              BUG_CHECK       UNKRSTRT,FATAL  ; Unknown restart code
               FEFF  007F                           .WORD   ^XFEFF
               0004' 0081                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_UNKRSTRT!4
                     0083   333 20$:     BUG_CHECK       IVLISTK,FATAL   ; Invalid interrupt stack (4)
               FEFF  0083                           .WORD   ^XFEFF
               0004' 0085                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_IVLISTK!4
                     0087   334 30$:     BUG_CHECK       DBLERR,FATAL    ; Double error halt (5)
               FEFF  0087                           .WORD   ^XFEFF
               0004' 0089                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_DBLERR!4
                     008B   335 40$:     BUG_CHECK       HALT,FATAL      ; Halt instruction (6)
               FEFF  008B                           .WORD   ^XFEFF
               0004' 008D                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_HALT!4
                     008F   336 50$:     BUG_CHECK       ILLVEC,FATAL    ; Illegal Vector code (7)
               FEFF  008F                           .WORD   ^XFEFF
               0004' 0091                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_ILLVEC!4
                     0093   337 60$:     BUG_CHECK       NOUSRWCS,FATAL  ; No user WCS for vector (8)
               FEFF  0093                           .WORD   ^XFEFF
               0004' 0095                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_NOUSRWCS!4
                     0097   338 70$:     BUG_CHECK       ERRHALT,FATAL   ; Error pending on halt (9)
               FEFF  0097                           .WORD   ^XFEFF
               0004' 0099                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_ERRHALT!4
                     009B   339 80$:     BUG_CHECK       CHMONIS,FATAL   ; CHM on interrupt stack (10)
               FEFF  009B                           .WORD   ^XFEFF
               0004' 009D                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_CHMONIS!4
                     009F   340 90$:     BUG_CHECK       CHMVEC,FATAL    ; CHM vector <1:0> .NE. 0 (11)
               FEFF  009F                           .WORD   ^XFEFF
               0004' 00A1                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_CHMVEC!4
                     00A3   341 100$:    BUG_CHECK       SCBRDERR,FATAL  ; SCB physical read error. (12)
               FEFF  00A3                           .WORD   ^XFEFF
               0004' 00A5                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_SCBRDERR!4
                     00A7   342 110$:    BUG_CHECK       WCSCORR,FATAL   ; WCS error correction failed (13)
               FEFF  00A7                           .WORD   ^XFEFF
               0004' 00A9                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_WCSCORR!4
                     00AB   343 120$:    BUG_CHECK       CPUCEASED,FATAL ; CPU ceased execution (14)
               FEFF  00AB                           .WORD   ^XFEFF
               0004' 00AD                           .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_CPUCEASED!4
                     00AF   344 130$:    BUG_CHECK       OUTOFSYNC,FATAL ; Processor clocks out of synch (15)
               FEFF  00AF                           .WORD   ^XFEFF
```

```
                        0004' 00B1                      .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_OUTOFSYNC!4
                              00B3    345 140$:   BUG_CHECK       ACCVIOMCHK      ; ACV or TNV during mchk exception (16)
                        FEFF  00B3                      .WORD   ^XFEFF
                        0000' 00B5                      .IIF DIF <CONT>,<FATAL> , .WORD BUG$_ACCVIOMCHK
                              00B7    346 150$:   BUG_CHECK       ACCVIOKSTK      ; ACV or TNV during kstk not valid (17)
                        FEFF  00B7                      .WORD   ^XFEFF
                        0000' 00B9                      .IIF DIF <CONT>,<FATAL> , .WORD BUG$_ACCVIOKSTK
                              00BB    347
                              00BB    348
                              00BB    349 POWERUP:                                    ;
                              00BB    350 ;
                              00BB    351 ; None of the interrupt stack area containing saved state will be overwritten
                              00BB    352 ; during the restart process in case another power failure occurs. The restart
                              00BB    353 ; procedure only reads the saved state and re-writes volatile registers so
                              00BB    354 ; that it can be repeated without harm.
                              00BB    355 ;
                   50   03    D0  00BB 356         MOVL    #CON$C_CLRWARM,R0       ; Console function=clear warm start flag.
                        52    D4  00BE 357         CLRL    R2                      ; Signal no return data expected.
             00000000'EF      16  00C0 358         JSB     CON$SENDCONSCMD         ; Send command to console.
        51   00000000'GF      D0  00C6 359         MOVL    G^EXE$GL_RPB,R1         ; Get virtual address of RPB.
          0C A1    01    CA  00CD 360         BICL    #1,RPB$L_RSTRTFLG(R1)   ; Clear flag to re-enable warmstart.
             00A4 C1    D5  00D1 361         TSTL    RPB$L_ISP(R1)           ; Test saved Interrupt SP from RPB.
                   0C    12  00D5 362         BNEQ    10$                     ; Branch if valid ISP.
                              00D7 363 ;
                              00D7 364 ; Interrupt stack pointer field in RPB is 0.  This indicates that the
                              00D7 365 ; the powerfail routine was not able to complete successfully, and that
                              00D7 366 ; it was unable to save the software state of the machine.
                              00D7 367 ;
      5E   51   00000200 8F    C1  00D7 368         ADDL3   #512,R1,SP              ; Use end of RPB for stack space.
                              00DF 369         BUG_CHECK -                     ; Fatal error.
                              00DF 370                 STATENTSVD,FATAL
                        FEFF  00DF                      .WORD   ^XFEFF
                        0004' 00E1                      .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_STATENTSVD!4
                              00E3    371 10$:
   15   00B4 C1    DA  00E3 372         MTPR    RPB$L_SISR(R1),#PR$_SISR; Restore software interrupt state.
   10   00A8 C1    DA  00E8 373         MTPR    RPB$L_PCBB(R1),#PR$_PCBB; Restore pointer to current PCB.
        0B   86    DA  00ED 374         MTPR    (R6)+,#PR$_P1LR         ; Restore P1 length register
        0A   86    DA  00F0 375         MTPR    (R6)+,#PR$_P1BR         ;  and P1 base register
        09   86    DA  00F3 376         MTPR    (R6)+,#PR$_P0LR         ; Restore real P0 length register
        08   86    DA  00F6 377         MTPR    (R6)+,#PR$_P0BR         ;  and P0 base register
        13   86    DA  00F9 378         MTPR    (R6)+,#PR$_ASTLVL       ; Restore AST level
        03   86    DA  00FC 379         MTPR    (R6)+,#PR$_USP          ; Restore user mode stack pointer
        02   86    DA  00FF 380         MTPR    (R6)+,#PR$_SSP          ; Restore supervisor mode stack pointer
        01   86    DA  0102 381         MTPR    (R6)+,#PR$_ESP          ; Restore executive mode stack pointer
        00   86    DA  0105 382         MTPR    (R6)+,#PR$_KSP          ; Restore kernel mode stack pointer
 00000000'EF      16  0108 383         JSB     EXE$REGRESTOR           ; Restore CPU-specific registers
                              010E 384 ;
                              010E 385 ; All saved Machine state has now been restored.  Renable SBI and CRD error
                              010E 386 ; interrupts, re-initialize interval timer and Scan device data base to
                              010E 387 ; set powerfail status for all units.  All controllers and devices are then
                              010E 388 ; re-initialized.
                              010E 389 ;
                   56   DD    010E 390         PUSHL   R6                      ; Save updated "stack pointer"
 00000000'EF      16  0110 391         JSB     EXE$INIPROCREG          ; Initialize processor registers
                              0116 392                                 ; for error detection and start interval
                              0116 393                                 ; timer.
                              0116 394 TIMERESET:                              ;
```

```
                 00000000'EF   16  0116   395          JSB     EXE$READ_TODR                   ; Get current time of day
        0000'CF  0004'CF   50   C1  011C   396          ADDL3   R0,W^EXE$GL_PWRINTVL,W^EXE$GL_PWRDONE ; Compute expected done
                                    0124   397                                                  ; time
        0000'CF  50   0000'CF   C3  0124   398          SUBL3   W^EXE$GL_PFAILTIM,R0,W^EXE$GL_PFATIM ; Get duration of power fail
                 50   0000'CF   C2  012C   399          SUBL    W^EXE$GL_TODR,R0                ; Compute time since boot
              50 50   1F   01   EF  0131   400          EXTZV   #1,#31,R0,R0                    ; Unsigned divide by 2
    50   00   50    00030D40 8F   7A  0136   401          EMUL    #<100*1000*2>,R0,#0,R0          ; Convert to 100 nanosecond units
                 50   0000'CF   C0  013F   402          ADDL    W^EXE$GQ_TODCBASE,R0            ; Compute current system time
                 51   0004'CF   D8  0144   403          ADWC    W^EXE$GQ_TODCBASE+4,R1          ;
                 0000'CF   50   7D  0149   404          MOVQ    R0,W^EXE$GQ_SYSTIME             ; Set as current system time
                 56   0000'CF   7E  014E   405          MOVAQ   W^LCK$GQ_BITMAP_EXP,R6          ; Get address of deadlock expiration
                           86   7C  0153   406          CLRQ    (R6)+                           ; timestamps and reset them
                           66   7C  0155   407          CLRQ    (R6)
                 56   0000'CF   9E  0157   408          MOVAB   W^EXE$GL_TQFL,R6                ; Get pointer to timer queue head
                      57   66   D0  015C   409          MOVL    (R6),R7                         ; Point at head of timer queue
                      57   56   D1  015F   410 10$:     CMPL    R6,R7                           ; Check for end of timer queue
                           17   13  0162   411          BEQL    30$                             ; Branch if yes
                   1C  A7   51   D1  0164   412          CMPL    R1,TQE$Q_TIME+4(R7)             ; Check high order bits for past due
                           0C   1F  0168   413          BLSSU   20$                             ; No try another
                           06   1A  016A   414          BGTRU   15$                             ; Past due, convert entry
                   18  A7   50   D1  016C   415          CMPL    R0,TQE$Q_TIME(R7)               ; High order bits equal, check low order
                           04   1F  0170   416          BLSSU   20$                             ; Not yet due
                   18  A7   50   7D  0172   417 15$:     MOVQ    R0,TQE$Q_TIME(R7)               ; Set new expiration time
                      57   67   D0  0176   418 20$:     MOVL    (R7),R7                         ; Flink to next entry
                           E4   11  0179   419          BRB     10$                             ;
                                    017B   420 30$:                                             ;
                 00000000'EF   16  017B   421          JSB     ERL$WARMSTART                   ; Log power recovery in the error log
                 00000000'GF   16  0181   422          JSB     G^CNX$POWER_FAIL                ; Inform connection manager of power recover
                                    0187   423                                                  ;  (This is an RSB if CLUSTRLOA is not loade
                                    0187   424 RESTARTIO:                                       ;
                      5D   5E   D0  0187   425          MOVL    SP,FP                           ; Save current stack pointer
              00000000'GF   16  018A   426          JSB     G^EXE$STARTUPADP                ; Call adapter initialization
                           5C   D4  0190   427          CLRL    AP                              ; Set up to
                      5C   01   AE  0192   428          MNEGW   #1,AP                           ; Initialize all controllers
                           28   10  0195   429          BSBB    EXE$INIT_DEVICE                 ; Call controller init routine
                                    0197   430
                      5E   5D   D0  0197   431          MOVL    FP,SP                           ; Restore stack pointer
                                    019A   432          SETIPL  #IPL$_POWER                     ; Block power fail interrupt
                      12   1F   DA  019A                MTPR    #IPL$_POWER,S^#PR$_IPL
                                    019D   433 ;
                                    019D   434 ; Drop IPL here to allow any impending powerfail interrupts to occur.  This
                                    019D   435 ; is because we have been running at IPL$_POWER, and if another powerfail
                                    019D   436 ; interrupt has occurred, it will be taken as soon as this routine REIs.
                                    019D   437 ; There would be no guarantee how much time the power down routine has left to
                                    019D   438 ; save the software state.  However, if we drop IPL BEFORE enabling subsequent
                                    019D   439 ; power fails, we allow any impending powerfail interrupt to occur; it will
                                    019D   440 ; essntially be ignored by the power down routine.  The power up routine will
                                    019D   441 ; then be re-executed.  And by the time we REI we are again guaranteed an
                                    019D   442 ; adequate amount of time to execute the power down routine.
                                    019D   443 ;
                                    019D   444          SETIPL  #<IPL$_POWER-2>                 ; Allow impending powerfail interrupts
                      12   1D   DA  019D                MTPR    #<IPL$_POWER-2>,S^#PR$_IPL
                           01   01A0   445          NOP                                     ; to occur before enabling another
                           01   01A1   446          NOP                                     ; execution of power down routine.
                                    01A2   447          SETIPL  #IPL$_POWER                     ; Back to guaranteed amount of time.
                      12   1F   DA  01A2                MTPR    #IPL$_POWER,S^#PR$_IPL
                      5E   8E   D0  01A5   448          MOVL    (SP)+,SP                        ; Set up to point to saved registers
```

```
                5D   00000000'GF    D0  01A8    449          MOVL    G^EXE$GL_RPB,FP          ; Get address of RPB.
                         1FFF 8F    BA  01AF    450          POPR    #^M<R0,R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,AP>
                         0000'CF    D4  01B3    451          CLRL    W^EXE$GL_PFAILTIM                ; Enable subsequent power fail
                         00A4 CD    D4  01B7    452          CLRL    RPB$L_ISP(FP)            ; Indicate software state not saved.
                              5D 8ED0  01BB    453          POPL    FP                       ; Restore FP.
                                 02    01BE    454          REI                              ; Return from powerfail restart.
                                       01BF    455
```

```
                                01BF    457          .SBTTL  EXE$INIT_DEVICE - Initialize device drivers
                                01BF    458
                                01BF    459 ;++
                                01BF    460 ; EXE$INIT_DEVICE - Call device drivers at controller and unit initialization
                                01BF    461 ;
                                01BF    462 ; INPUTS:
                                01BF    463 ;
                                01BF    464 ;       Low order word:
                                01BF    465 ;         AP = -1  -> Do initialization for all devices on all adaptors
                                01BF    466 ;         AP >= 0  -> Only initialize for devices on this TR level
                                01BF    467 ;
                                01BF    468 ;       Hi order word:
                                01BF    469 ;         AP = -1  -> Called from INIT - No powerfail
                                01BF    470 ;         AP =  0  -> Called from POWERFAIL/ADAPTERR (UBA powerfail)
                                01BF    471 ;
                                01BF    472 ; OUTPUTS:
                                01BF    473 ;
                                01BF    474 ;       Device controller and units initialized
                                01BF    475 ;       All registers destroyed!!!!!
                                01BF    476 ;--
                                01BF    477
                                01BF    478 EXE$INIT_DEVICE::
                                01BF    479
                          5B    D4    01BF    480          CLRL    R11                     ; Initial condition
                                01C1    481
00000000'GF                     16    01C1    482 DDBLOOP:JSB     G^IOC$SCAN_IODB         ; Scan the I/O data base
                          01 50 E8    01C7    483          BLBS    R0,5$                   ; Found another UCB
                                05    01CA    484          RSB                             ; Thats all, return
                                01CB    485
                          5C    D5    01CB    486 5$:      TSTL    AP                      ; Check if POWERFAIL mode
                                06    18    01CD    487          BGEQ    7$                      ; Yes, skip next
00000000'GF                     16    01CF    488          JSB     G^IOC$RELOC_DDT         ; Make offsets absolute system addresses
                       5A   D4 AB    DE    01D5    489 7$:      MOVAL   DDB$L_UCB-UCB$L_LINK(R11),R10   ; Get address of first UCB address
                                58    D4    01D9    490          CLRL    R8                      ; Clear last CRB address
                       5A   30 AA    D0    01DB    491 10$:     MOVL    UCB$L_LINK(R10),R10     ; Get address of next UCB
                                E0    13    01DF    492          BEQL    DDBLOOP                 ; If zero, no more for this DDB
                F5 38 AA        14    E0    01E1    493          BBS     S^#DEV$V_MBX,UCB$L_DEVCHAR(R10),10$ ; Branch if mailbox
                                      01E6    494 ;
                                      01E6    495 ; Check to see if we must init all devices on all adaptors or on just
                                      01E6    496 ; one specific adaptor.
                                      01E6    497 ;
                       54   24 AA    D0    01E6    498          MOVL    UCB$L_CRB(R10),R4       ; Point to CRB
                                5C    B5    01EA    499          TSTW    AP                      ; If AP neg, init all
                                0C    19    01EC    500          BLSS    15$                     ; Init all
                       50   38 A4    D0    01EE    501          MOVL    CRB$L_INTD+VEC$L_ADP(R4),R0 ; Point to ADP
                                E7    13    01F2    502          BEQL    10$                     ; No adaptor for this "device"
                    0C A0       5C    B1    01F4    503          CMPW    AP,ADP$W_TR(R0)         ; TR's match
                                E1    12    01F8    504          BNEQ    10$                     ; No, look for others
                                      01FA    505
                                5C    D5    01FA    506 15$:     TSTL    AP                      ; Check if POWERFAIL mode
                                04    19    01FC    507          BLSS    17$                     ; No, do not set it
                    64 AA       20    A8    01FE    508          BISW    #UCB$M_POWER,UCB$W_STS(R10)     ; Set power failed status
                          58    54    D1    0202    509 17$:     CMPL    R4, R8                  ; Is this the same CRB?
                                0E    13    0205    510          BEQL    40$                     ; Branch if same CRB.
                          58    54    D0    0207    511          MOVL    R4, R8                  ; Save new CRB address.
                                51    D4    020A    512          CLRL    R1                      ; We have no extra CSR info
                                      020C    513                                          ; (SYSGEN does).
```

```
00000000'GF   16  020C   514          JSB     G^IOC$CTRLINIT          ; Do driver controller initialization.
      41 50   E9  0212   515          BLBC    R0, 70$                 ; Branch if CSR test failed.
   55   5A   D0  0215   516 40$:     MOVL    R10, R5                 ; Setup UCB address.
00000000'GF   16  0218   517          JSB     G^IOC$UNITINIT          ; Do driver unit initialization.
      03   B3  021E   518          BITW    #UCB$M_INT!UCB$M_TIM,-
    64 A5         0220   519                  UCB$W_STS(R5)           ; Interrupt or timeout expected?
      B7   13  0222   520          BEQL    10$                     ; If eql then no
64 A5   02   AA  0224   521          BICW    #UCB$M_INT,UCB$W_STS(R5); Clear interrupt expected
64 A5   01   A8  0228   522          BISW    #UCB$M_TIM,UCB$W_STS(R5); Set timeout expected
    6C A5    D4  022C   523          CLRL    UCB$L_DUETIM(R5)        ; Now
                022F   524 ;
                022F   525 ; Look for busy, non-MSCP disks that are not in mount verification.  Clear
                022F   526 ; volume-valid and set mount-verification-pending so that restarted I/Os will
                022F   527 ; fail and the volume will be revalidated.  Non-busy disks are handled
                022F   528 ; independently.
                022F   529 ;
    40 A5    91  022F   530          CMPB    UCB$B_DEVCLASS(R5),-    ; Make sure it is a disk
       01         0232   531                  #DC$_DISK
         A6   12  0233   532          BNEQ    10$
         0E   E1  0235   533          BBC     #DEV$V_FOD,-            ; Not file oriented
 A1 38 A5         0237   534                  UCB$L_DEVCHAR(R5),10$
         05   E0  023A   535          BBS     #DEV$V_SQD,-            ; Sequential device
 9C 38 A5         023C   536                  UCB$L_DEVCHAR(R5),10$
         05   E0  023F   537          BBS     #DEV$V_MSCP,-           ; MSCP disks are handled independently
 97 3C A5         0241   538                  UCB$L_DEVCHAR2(R5),10$
         0E   E2  0244   539          BBSS    #UCB$V_MNTVERIP,-       ; Mount verification already in progress
 92 64 A5         0246   540                  UCB$L_STS(R5),10$
         13   E2  0249   541          BBSS    #UCB$V_MNTVERPND,-      ; Mark it mount verification pending
 00 64 A5         024B   542                  UCB$L_STS(R5),50$
         0B   E5  024E   543 50$:     BBCC    #UCB$V_VALID,-          ; Cause I/O to fail
 00 64 A5         0250   544                  UCB$W_STS(R5),51$
      FF85    31  0253   545 51$:     BRW     10$                     ; Next unit
                0256   546
         59   D4  0256   547 70$:     CLRL    R9                      ; Zap CRB to force CRB search
64 AA    10   AA  0258   548          BICW    #UCB$M_ONLINE,UCB$W_STS(R10)    ; Set unit offline
      FF7C    31  025C   549          BRW     10$                     ; Continue search
```

```
                                                J 8
                     025F    551          .SBTTL  EXE$PWRTIMCHK - Check for reasonable interval since power recovery
                     025F    552 ;++
                     025F    553 ; Functional Description:
                     025F    554 ;       EXE$PWRTIMCHK is called by driver initialization code to check for
                     025F    555 ;       a sufficient interval since power recovery to expect devices to be
                     025F    556 ;       ready again.  If the return from EXE$PWRTIMCHK indicates that the
                     025F    557 ;       reasonable interval has not yet elapsed, the device driver may elect
                     025F    558 ;       to wait for a while using EXE$PWRTIMCHK check the time.
                     025F    559 ;
                     025F    560 ; Calling Sequence:
                     025F    561 ;       BSB/JSB EXE$PWRTIMCHK
                     025F    562 ;
                     025F    563 ; Output Parameters:
                     025F    564 ;       R0 - Low bit clear if interval expired.
                     025F    565 ;--
                     025F    566 EXE$PWRTIMCHK::                                     ;
 00000000'EF   16    025F    567          JSB     EXE$READ_TODR                      ; Get current time of day
      7E   50   D0  0265    568          MOVL    R0,-(SP)                           ; Save it temporarily
           50   D4  0268    569          CLRL    R0                                 ; Assume interval expired
8E    0000'CF   D1  026A    570          CMPL    W^EXE$GL_PWRDONE,(SP)+             ; Check for time expired
           02   1B  026F    571          BLEQU   10$                                ; Exit with low bit clear if expired
           50   D6  0271    572          INCL    R0                                 ; Else set low bit of R0
                05  0273    573 10$:     RSB                                        ;
                    0274    574
                    0274    575          .END                                       ;
```

```
ADP$W_TR                        = 0000000C
BUG$_ACCVIOKSTK                   ********    X   04
BUG$_ACCVIOMCHK                   ********    X   04
BUG$_CHMONIS                      ********    X   04
BUG$_CHMVEC                       ********    X   04
BUG$_CPUCEASED                    ********    X   04
BUG$_DBLERR                       ********    X   04
BUG$_ERRHALT                      ********    X   04
BUG$_HALT                         ********    X   04
BUG$_ILLVEC                       ********    X   04
BUG$_IVLISTK                      ********    X   04
BUG$_NOUSRWCS                     ********    X   04
BUG$_OUTOFSYNC                    ********    X   04
BUG$_SCBRDERR                     ********    X   04
BUG$_STATENTSVD                   ********    X   04
BUG$_UNKRSTRT                     ********    X   04
BUG$_WCSCORR                      ********    X   04
CNX$POWER_FAIL                    ********    X   04
CON$C_CLRWARM                   = 00000003
CON$SENDCONSCMD                   ********    X   04
CRB$L_INTD                      = 00000024
DC$_DISK                        = 00000001
DDB$L_UCB                       = 00000004
DDBLOOP                           000001C1 R      04
DEV$V_FOD                       = 0000000E
DEV$V_MBX                       = 00000014
DEV$V_MSCP                      = 00000005
DEV$V_SQD                       = 00000005
ERL$WARMSTART                     ********    X   04
EXE$GL_PFAILTIM                   ********    X   03
EXE$GL_PFATIM                     ********    X   04
EXE$GL_PWRDONE                    00000000 RG     02
EXE$GL_PWRINTVL                   00000004 RG     02
EXF$GL_RPB                        ********    X   03
EXE$GL_TODR                       ********    X   04
EXE$GL_TQFL                       ********    X   04
EXE$GQ_SYSTIME                    ********    X   04
EXE$GQ_TODCBASE                   ********    X   04
EXE$INIPROCREG                    ********    X   04
EXE$INIT_DEVICE                   000001BF RG     04
EXE$POWERFAIL                     00000000 RG     03
EXE$PWRTIMCHK                     0000025F RG     04
EXE$READ_TODR                     ********    X   03
EXE$REGRESTOR                     ********    X   04
EXE$REGSAVE                       ********    X   03
EXE$RESTART                       00000000 RG     04
EXE$STARTUPADP                    ********    X   04
IOC$CTRLINIT                      ********    X   04
IOC$RELOC_DDT                     ********    X   04
IOC$SCAN_IODB                     ********    X   04
IOC$UNITINIT                      ********    X   04
IPL$_POWER                      = 0000001F
LCK$GQ_BITMAP_EXP                 ********    X   04
POWERUP                           000000BB R      04
PR$_ASTLVL                      = 00000013
PR$_ESP                         = 00000001
PR$_IPL                         = 00000012
PR$_KSP                         = 00000000
PR$_MAPEN                       = 00000038
PR$_P0BR                        = 00000008
PR$_P0LR                        = 00000009
PR$_P1BR                        = 0000000A
PR$_P1LR                        = 0000000B
PR$_PCBB                        = 00000010
PR$_SBR                         = 0000000C
PR$_SCBB                        = 00000011
PR$_SISR                        = 00000015
PR$_SLR                         = 0000000D
PR$_SSP                         = 00000002
PR$_TBIA                        = 00000039
PR$_USP                         = 00000003
RESTARTIO                         00000187 R      04
RESTRT_CHM                      = 0000000A
RESTRT_DBLERR                   = 00000005
RESTRT_ERRHALT                  = 00000009
RESTRT_HALT                     = 00000006
RESTRT_ILLVEC                   = 00000007
RESTRT_IVLISTK                  = 00000004
RESTRT_NOUSRWCS                 = 00000008
RESTRT_POWERUP                  = 00000003
RPB$L_ISP                       = 000000A4
RPB$L_PCBB                      = 000000A8
RPB$L_RSTRTFLG                  = 0000000C
RPB$L_SBR                       = 000000AC
RPB$L_SCBB                      = 000000B0
RPB$L_SISR                      = 000000B4
RPB$L_SLR                       = 000000B8
RPB$L_SVASPT                    = 00000050
TIMERESET                         00000116 R      04
TQE$Q_TIME                      = 00000018
UCB$B_DEVCLASS                  = 00000040
UCB$L_CRB                       = 00000024
UCB$L_DEVCHAR                   = 00000038
UCB$L_DEVCHAR2                  = 0000003C
UCB$L_DUETIM                    = 0000006C
UCB$L_LINK                      = 00000030
UCB$L_STS                       = 00000064
UCB$M_INT                       = 00000002
UCB$M_ONLINE                    = 00000010
UCB$M_POWER                     = 00000020
UCB$M_TIM                       = 00000001
UCB$V_MNTVERIP                  = 0000000E
UCB$V_MNTVERPND                 = 00000013
UCB$V_VALID                     = 0000000B
UCB$W_STS                       = 00000064
VEC$L_ADP                       = 00000014
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
$$$220	00000008 (8.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
$AEXENONPAGED	0000005B (91.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
$AAEXENONPAGED	00000274 (628.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC PAGE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.06	00:00:01.77
Command processing	119	00:00:00.61	00:00:05.18
Pass 1	346	00:00:11.41	00:00:34.23
Symbol table sort	0	00:00:01.69	00:00:04.83
Pass 2	135	00:00:02.49	00:00:09.81
Symbol table output	12	00:00:00.09	00:00:00.71
Psect synopsis output	3	00:00:00.02	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	652	00:00:16.37	00:00:56.56

The working set limit was 1650 pages.
68206 bytes (134 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1163 non-local and 33 local symbols.
575 source lines were read in Pass 1, producing 20 object records in Pass 2.
26 pages of virtual memory were used to define 25 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	16
_255DUA28:[SYSLIB]STARLET.MLB;2	6
TOTALS (all libraries)	22

1256 GETS were required to define 22 macros.

There were no errors, warnings or information messages.

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

0379 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RELOCDRV
LIS

RMSRESET
LIS

PROCSTRT
LIS

PHDUTL
LIS

PMSDAT
LIS

POSTEF
LIS

POWERFAIL
LIS

PRDEF
LIS

PTEDUMP
LIS

PDAT
LIS

RMSVECTOR
LIS