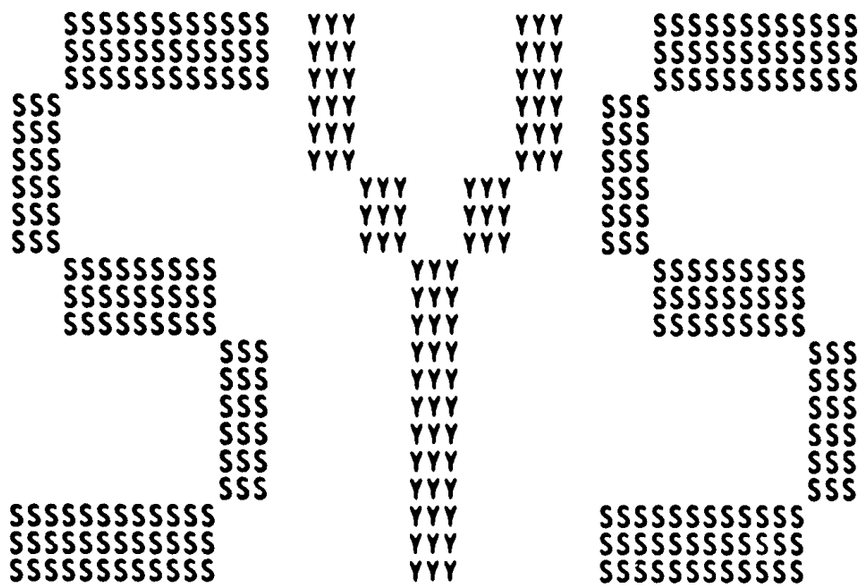

FILEID**TIMESCHDL

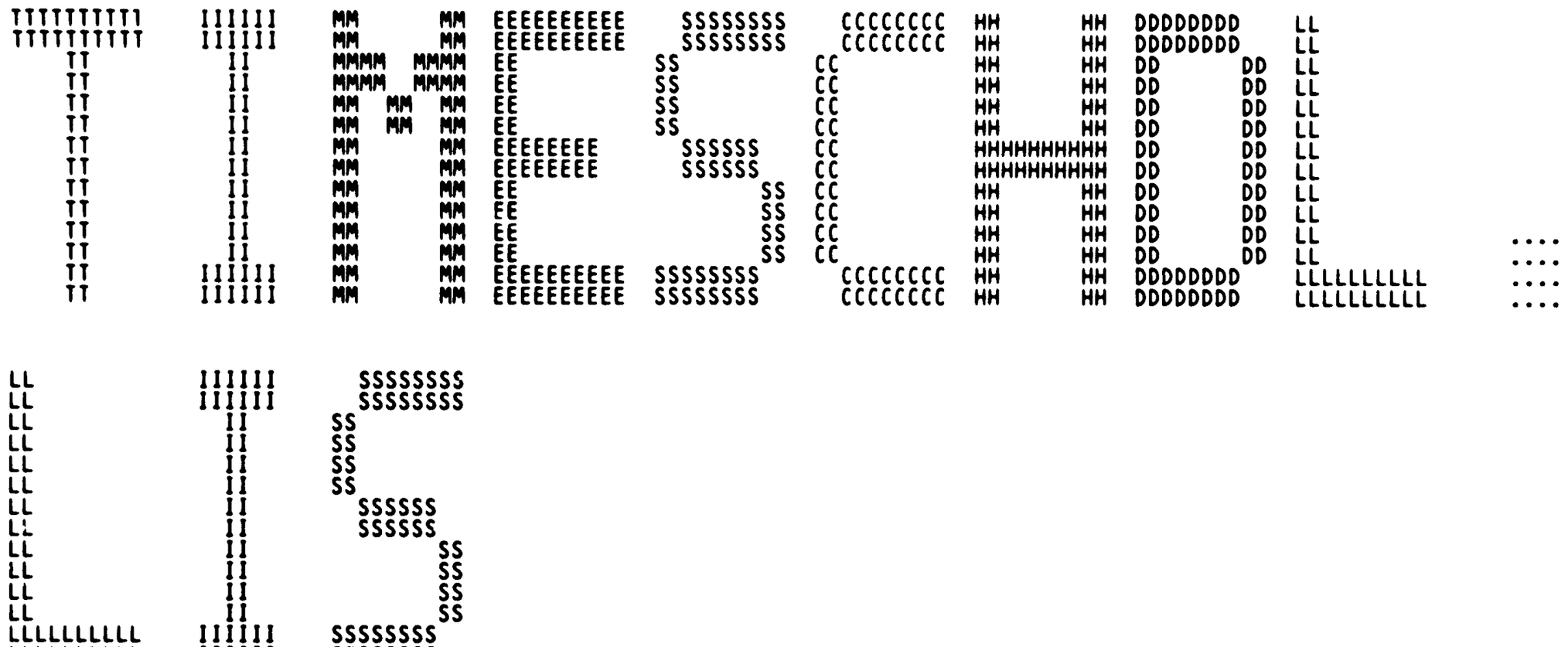

Table of contents

```
0000     1          .TITLE  TIMESCHDL - TIME DEPENDENT SCHEDULING
0000     2          .IDENT  'V04-000'
0000     3
0000     4  ;
0000     5  ;**********************************************************************
0000     6  ;*                                                                    *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
0000     9  ;*  ALL RIGHTS RESERVED.                                              *
0000    10  ;*                                                                    *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
0000    12  ;*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE     *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER  *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
0000    15  ;*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY  *
0000    16  ;*  TRANSFERRED.                                                       *
0000    17  ;*                                                                    *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE  *
0000    19  ;*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20  ;*  CORPORATION.                                                       *
0000    21  ;*                                                                    *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS  *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.           *
0000    24  ;*                                                                    *
0000    25  ;*                                                                    *
0000    26  ;**********************************************************************
0000    27  ;
0000    28  ; D. N. CUTLER 15-JUN-76
0000    29  ;
0000    30  ; TIME RELATED ACTIVITY
0000    31  ;
0000    32  ;       UPDATE TIME OF DAY,
0000    33  ;       CHECK FOR ITEM READY IN TIMER QUEUE, AND
0000    34  ;       PERFORM DEVICE TIMEOUT.
0000    35  ;       UPDATE MEASUREMENT STATISTICS IF MEASUREMENT IS ENABLED.
0000    36  ;
0000    37  ; MODIFICATION HISTORY:
0000    38  ;
0000    39  ;       V03-013 CWH3012         CW Hobbs                29-Apr-1984
0000    40  ;               Change all W^SCH$Gxxx references to L^SCH$Gxxx to
0000    41  ;               fix current (and future) branch problems.
0000    42  ;
0000    43  ;       V03-012 WMC0012         Wayne Cardoza           23-Apr-1984
0000    44  ;               Declare pagedmeory and mailbox resources available once per
0000    45  ;               second just in case...
0000    46  ;
0000    47  ;       V03-011 CWH3011         CW Hobbs                14-Apr-1984
0000    48  ;               Fixed broken branch.
0000    49  ;
0000    50  ;       V03-010 SRB0118         Steve Beckhardt         26-Mar-1984
0000    51  ;               Fixed broken branches.
0000    52  ;
0000    53  ;       V03-009 SRB0117         Steve Beckhardt         17-Mar-1984
0000    54  ;               Removed loop around checking for locks on the timeout queue
0000    55  ;               needing a deadlock search.  This loop is now in the module
0000    56  ;               DEADLOCK.
0000    57  ;
```

```
0000     58 ;       V03-008 MIR0200         MICHAEL I. ROSENBLUM    15-OCT-1983
0000     59 ;               Remove the setipl to DIPL when RDUTIM expires
0000     60 ;               To allow the driver using this service to do it's
0000     61 ;               own synchronization.
0000     62 ;       V03-007 DWT0123         David W. Thiel          22-Aug-1983
0000     63 ;               Declare non-paged dynamic memory available once per
0000     64 ;               second.
0000     65 ;
0000     66 ;       V03-006 SRB0099         Steve Beckhardt         15-July-1983
0000     67 ;               Added loop to deadlock detection timeout code to allow
0000     68 ;               finding more than one deadlock per second.
0000     69 ;
0000     70 ;       V03-005 ROW0190         Ralph O. Weber           3-MAY-1983
0000     71 ;               Rewrite fork-and-wait processing using newly acquired
0000     72 ;               knowledge of how to move a queue from one header to another.
0000     73 ;               This better protects against infinite looping during the
0000     74 ;               processing of the fork-and-wait queue.
0000     75 ;
0000     76 ;       V03-004 ROW0176         Ralph O. Weber           4-APR-1983
0000     77 ;               Add code to process fork-and-wait queue to EXE$TIMEOUT.
0000     78 ;
0000     79 ;       V03-003 SRB0056         Steve Beckhardt         14-Dec-1982
0000     80 ;               Modified code to allow IPL$_SYNCH and IPL$_TIMER to be IPL 8
0000     81 ;               while IPL$_TIMERFORK is IPL 7.
0000     82 ;
0000     83 ;       V03-002 SPF0200         Steve Forgey            1-Jun-1982
0000     84 ;               Add alternate entry point to interval timer interrupt service
0000     85 ;               routine for Unibus clock interrupt.
0000     86 ;
0000     87 ;       V03-001 ROW0078         Ralph O. Weber          1-APR-1982
0000     88 ;               Enhance documentation regarding multiple use of timer queue
0000     89 ;               entry blocks.
0000     90 ;
0000     91 ;
```

```
0000     93          .SBTTL  DECLARATIONS
0000     94 ;
0000     95 ; MACRO LIBRARY CALLS
0000     96 ;
0000     97
0000     98          $ACBDEF                         ;DEFINE ACB OFFSETS
0000     99          $CADEF                          ;DEFINE CONDITIONAL ASSEMBLY PARAMETERS
0000    100          $CRBDEF                         ;DEFINE CRB OFFSETS
0000    101          $DDBDEF                         ;DEFINE DDB OFFSETS
0000    102          $DEVDEF                         ;DEFINE DEVICE CHARACTERISTICS
0000    103          $ERLDEF                         ;DEFINE ERL OFFSETS
0000    104          $FKBDEF                         ;DEFINE FORK BLOCK OFFSETS
0000    105          $IPLDEF                         ;DEFINE INTERRUPT PRIORITY LEVELS
0000    106          $JIBDEF                         ;DEFINE JIB OFFSETS
0000    107          $LKBDEF                         ;DEFINE LKB OFFSETS
0000    108          $PCBDEF                         ;DEFINE PCB OFFSETS
0000    109          $PHDDEF                         ;DEFINE PHD OFFSETS
0000    110          $PRDEF                          ;DEFINE PROCESSOR REGISTERS
0000    111          $PRIDEF                         ;DEFINE PRIORITY INCREMENTS
0000    112          $PSLDEF                         ;DEFINE PROCESSOR STATUS FIELDS
0000    113          $RSNDEF                         ;DEFINE RESOURCE WAIT NUMBERS
0000    114          $STATEDEF                       ;DEFINE SCHEDULER STATE VALUES
0000    115          $TQEDEF                         ;DEFINE TQE OFFSETS
0000    116          $UCBDEF                         ;DEFINE UCB OFFSETS
0000    117          $TTYDEFS                        ; TTY UCB extension (must FOLLOW $UCBDEF)
0000    118          $TTYDEF                         ;DEFINE TTY SYMBOLS
```

```
                                0000   120          .SBTTL  HARDWARE CLOCK INTERRUPTS
                                0000   121 ;+
                                0000   122 ; EXE$HWCLKINT - HARDWARE CLOCK INTERRUPT
                                0000   123 ;
                                0000   124 ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN THE CLOCK COUNT REGISTER
                                0000   125 ; OVERFLOWS. THE CURRENT ABSOLUTE TIME IS UPDATED, THE ACCOUNTING INTEGRAL
                                0000   126 ; OF THE CURRENT PROCESS IS INCREMENTED, AND THE QUANTUM OF THE CURRENT
                                0000   127 ; PROCESS IS INCREMENTED. IF THE PROCESS QUANTUM TRANSISTS TO ZERO OR THE
                                0000   128 ; FIRST ENTRY IN THE TIMER QUEUE CAN BE REMOVED, THEN A SOFTWARE INTERRUPT
                                0000   129 ; IS REQUESTED ON THE TIMER LEVEL. THE CLOCK INTERRUPT IS THEN DISMISSED.
                                0000   130 ;
                                0000   131 ;       THE CLOCK IS CURRENTLY SET TO INTERRUPT AT 10MS INTERVALS.
                                0000   132 ;-
                                0000   133
                           00000000    134          .PSECT  A$EXENONPAGED,LONG
                                0000   135 EXE$HWCLKINT::                            ;HARDWARE CLOCK INTERRUPT
        18   800000C1 8F     DA 0000   136          MTPR    #^X800000C1,#PR$_ICCS   ;CLEAR INTERRUPT + ERROR AND RE-ENABLE
                             01 0007   137          NOP                             ;FORCE NEXT INST. TO LONGWORD ALIGN
                                0008   138 EXE$UBCLKINT::
                          50    DD 0008   139          PUSHL   R0                      ;SAVE REGISTER R0
    0000'CF    000186A0 8F     C0 000A   140          ADDL    #100000,W^EXE$GQ_SYSTIME ;UPDATE SYSTEM ABSOLUTE TIME
            0004'CF    00     D8 0013   141          ADWC    #0,W^EXE$GQ_SYSTIME+4   ;
                                0018   142
                                0018   143 ;
                                0018   144 ; IF MEASUREMENT IS ENABLED, UPDATE TIMER STATISTICS FIELDS.
                                0018   145 ;
                                0018   146
                           00000002    147          .IF NE  CA$_MEASURE
                                0018   148
          50    0B AE     98 0018   149          CVTBL   11(SP),R0               ;GET UPPER BYTE OF SAVED PSL
                03     18 001C   150          BGEQ    20$                     ;BRANCH IF CM BIT NOT SET
             50    05     9A 001E   151          MOVZBL  #5,R0                   ;ELSE INSERT CM INDEX
          50    F8 8F     8A 0021   152 20$:     BICB    #^XF8,R0                ;CONVERT EXTENDED BYTE TO INDEX
           0000'CF40     D6 0025   153          INCL    W^PMS$GL_KERNEL[R0]     ;INCREMENT STATISTICS VECTOR
                                002A   154
                                002A   155          .ENDC
                                002A   156
       13 08 AE     1A     E0 002A   157          BBS     #PSL$V_IS,8(SP),25$     ;IF SET, DON'T ADD INTERRUPT SERVICE TIME
    50    00000000'EF     D0 002F   158          MOVL    L^SCH$GL_CURPCB,R0      ;GET ADDRESS OF CURRENT PROCESS PCB
          50    6C     A0 D0 0036   159          MOVL    PCB$L_PHD(R0),R0        ;GET ADDRESS OF PROCESS HEADER
                38     A0 D6 003A   160          INCL    PHD$L_CPUTIM(R0)        ;INCREMENT ACCOUNTING INTEGRAL
                3C     A0 B6 003D   161          INCW    PHD$W_QUANT(R0)         ;INCREMENT TIME QUANTUM
                   17     18 0040   162          BGEQ    30$                     ;IF GEQ, QUANTUM RUNOUT
                                0042   163 25$:
          50    0000'CF     D0 0042   164          MOVL    W^EXE$GL_TQFL,R0        ;GET ADDRESS OF FIRST ENTRY IN TIME QUEUE
    0004'CF    1C     A0 D1 0047   165          CMPL    TQE$Q_TIME+4(R0),W^EXE$GQ_SYSTIME+4 ;COMPARE HIGH ORDER PARTS OF TIM
                   0A     1F 004D   166          BLSSU   30$                     ;IF LSSU ENTRY DUE
                   0B     1A 004F   167          BGTRU   40$                     ;IF GTRU ENTRY NOT DUE
    0000'CF    18     A0 D1 0051   168          CMPL    TQE$Q_TIME(R0),W^EXE$GQ_SYSTIME ;COMPARE LOW ORDER PARTS OF TIME
                   03     1A 0057   169          BGTRU   40$                     ;IF GTRU ENTRY NOT DUE
                                0059   170 30$:     SOFTINT #IPL$_TIMERFORK         ;REQUEST SOFTWARE INTERRUPT ON TIMER LEVEL
                50 8ED0     005C   171 40$:     POPL    R0                      ;RESTORE R0
                        02     005F   172          REI                             ;
```

```
                                0060   174          .SBTTL  SOFTWARE TIMER INTERRUPTS
                                0060   175 ;+
                                0060   176 ; EXE$SWTIMINT - SOFTWARE TIMER INTERRUPTS
                                0060   177 ;
                                0060   178 ; THIS ROUTINE IS AUTOMATICALLY VECTORED TO WHEN A SOFTWARE INTERRUPT IS
                                0060   179 ; REQUESTED ON THE TIMER LEVEL. TIMER INTERRUPTS ARE REQUESTED WHEN THE
                                0060   180 ; CURRENT PROCESS HAS EXCEEDED ITS CPU TIME QUANTUM OR THE FIRST ENTRY IN
                                0060   181 ; THE TIMER QUEUE IS DUE.
                                0060   182 ;-
                                0060   183          ASSUME  IPL$_TIMER EQ IPL$_SYNCH
                                0060   184
                                0060   185          .ALIGN  LONG
                                0060   186 EXE$SWTIMINT::                              ;SOFTWARE TIMER INTERRUPTS
                                0060   187          SETIPL  #IPL$_TIMER                ;RAISE IPL TO TIMER/SYNCH
                          3F    BB    0063   188          PUSHR   #^M<R0,R1,R2,R3,R4,R5>     ;SAVE REGISTERS R0 THRU R5
54    00000000'EF             D0    0065   189          MOVL    L^SCH$GL_CURPCB,R4         ;GET CURRENT PROCESS PCB ADDRESS
     55    6C A4              D0    006C   190          MOVL    PCB$L_PHD(R4),R5           ;GET ADDRESS OF PROCESS HEADER
           3C A5              B5    0070   191          TSTW    PHD$W_QUANT(R5)            ;QUANTUM END?
             03               19    0073   192          BLSS    CHKTMQ                     ;IF LSS NO
          FF88'               30    0075   193          BSBW    SCH$QEND                   ;CALL SCHEDULER TO RESET QUANTUM
   55    0000'CF              D0    0078   194 CHKTMQ: MOVL    W^EXE$GL_TQFL,R5           ;GET ADDRESS OF FIRST ENTRY IN TIME QUEUE
                                007D   195          SETIPL  #IPL$_HWCLK                ;RAISE IPL TO HARDWARE CLOCK LEVEL
0004'CF    1C A5              D1    0080   196          CMPL    TQE$Q_TIME+4(R5),W^EXE$GQ_SYSTIME+4 ;COMPARE HIGH ORDER PARTS OF TIM
             10               1F    0086   197          BLSSU   20$                        ;IF LSSU ENTRY IS DUE
             08               1A    0088   198          BGTRU   10$                        ;IF GTRU ENTRY IS NOT DUE
0000'CF    18 A5              D1    008A   199          CMPL    TQE$Q_TIME(R5),W^EXE$GQ_SYSTIME ;COMPARE LOW ORDER PARTS OF TIME
             06               1B    0090   200          BLEQU   20$                        ;IF LEQU ENTRY IS DUE
             3F               BA    0092   201 10$:     POPR    #^M<R0,R1,R2,R3,R4,R5>     ;RESTORE REGISTERS R0 THRU R5
                              0094   202          SETIPL  #IPL$_TIMERFORK            ;LOWER IPL BACK TO TIMERFORK
             02               0097   203          REI                                ;
                              0098   204
                              0098   205 ;
                              0098   206 ; REMOVE DUE ENTRY FROM TIMER QUEUE
                              0098   207 ;
                              0098   208
          55    65    0F      0098   209 20$:     REMQUE  (R5),R5                    ;REMOVE FIRST ENTRY FROM TIME QUEUE
                              009B   210          SETIPL  #IPL$_TIMER                ;LOWER IPL TO SOFTWARE TIMER LEVEL
50  0B A5     02    00    EF  009E   211          EXTZV   #0,#2,TQE$B_RQTYPE(R5),R0  ;GET REQUEST TYPE
                              00A4   212          CASE    R0,<TIMER,SYSUB,WAKEUP>    ;DISPATCH TO PROCESSING ROUTINE
                              00AE   213          BUG_CHECK INVTQEFMT                ;INVALID TIME QUEUE ENTRY FORMAT
             C4               11    00B2   214          BRB     CHKTMQ                     ;
                              00B4   215
                              00B4   216 ;
                              00B4   217 ; PROCESS SYSTEM SUBROUTINE
                              00B4   218 ;
                              00B4   219 ; **** WARNING ****
                              00B4   220 ;
                              00B4   221 ; Upon return from the system subroutine call, this routine expects R5 to
                              00B4   222 ; contain the address of a valid timer queue entery.  The TQE$V_REPEAT bit
                              00B4   223 ; of that TQE will be tested, and if it is set, the TQE will be reentered in
                              00B4   224 ; the timer queue.  Therefore, the call system subroutine CANNOT use the TQE,
                              00B4   225 ; pointed to by R5 at entry, for some other purpose and return here without
                              00B4   226 ; placing the address of a valid TQE in R5.
                              00B4   227 ;
                              00B4   228 ; To this end, the executive system table contains a global symbol,
                              00B4   229 ; EXE$AL_TQENOREPT, which repersents the address of a always valid always
                              00B4   230 ; non-repeating timer queue entry.  Timer system subroutines wishing to use
```

```
                                00B4   231 ; the TQE which caused them to be called for purposes other than continued
                                00B4   232 ; use of the timer queue may load the address represented by EXE$AL_TQENOREPT
                                00B4   233 ; into R5 before returing to this routine.  This prevents duplicate use of
                                00B4   234 ; the TQE block which resulted in the system subroutine being called.
                                00B4   235 ;
                                00B4   236 ; For example, a system subroutine which decides to discontinue its timed
                                00B4   237 ; operations and deallocate the TQE would execute at least the following
                                00B4   238 ; instructions:
                                00B4   239 ;               ...
                                00B4   240 ;       MOVL    R5, R0                  ;NB: this uses the TQE
                                00B4   241 ;       JSB     G^COM$DRVDEALMEM        ;    as an IPL 6 fork block.
                                00B4   242 ;               ...
                                00B4   243 ;       MOVAL   G^EXE$AL_TQENOREPT, R5  ;Setup no repeat TQE.
                                00B4   244 ;       RSB
                                00B4   245
    53    10 A5    7D          00B4   246 SYSUB:  MOVQ    TQE$L_FR3(R5),R3        ;LOAD SUBROUTINE CONTEXT
          0C B5    16          00B8   247         JSB     @TQE$L_FPC(R5)          ;CALL SYSTEM SUBROUTINE
B8 0B A5    02    E1          00BB   248         BBC     #TQE$V_REPEAT,TQE$B_RQTYPE(R5),CHKTMQ ;IF CLR, NOT REPEATABLE
              60    11          00C0   249         BRB     REPTIM                  ;INSERT REPEAT REQUEST IN TIME QUEUE
                                00C2   250
                                00C2   251 ;
                                00C2   252 ; PROCESS TIMER
                                00C2   253 ;
                                00C2   254
                                00C2   255         .ENABL  LSB
    51    0C A5    D0          00C2   256 TIMER:  MOVL    TQE$L_PID(R5),R1        ;GET TARGET PROCESS ID
       52    02    9A          00C6   257         MOVZBL  #PRI$_TIMER,R2          ;SET PRIORITY INCREMENT CLASS
    53    29 A5    9A          00C9   258         MOVZBL  TQE$B_EFN(R5),R3        ;GET EVENT FLAG NUMBER
          FF30'    30          00CD   259         BSBW    SCH$POSTEF              ;POST EVENT FLAG
          38 50    E9          00D0   260         BLBC    R0,30$                  ;IF LBC PROCESS NO LONGER IN SYSTEM
 50    0080 C4    D0          00D3   261         MOVL    PCB$L_JIB(R4),R0        ;GET JIB ADDRESS
          34 A0    B6          00D8   262         INCW    JIB$W_TQCNT(R0)         ;UPDATE AVAILABLE TIME QUEUE ENTRIES
    25 28 A5    06    E1          00DB   263         BBC     #ACB$V_QUOTA,TQE$B_RMOD(R5),20$ ;IF CLR, NO AST SPECIFIED
    0B A5    28 A5    90          00E0   264         MOVB    TQE$B_RMOD(R5),TQE$B_RQTYPE(R5) ;SET AST ACCESS MODE
          52    02    9A          00E5   265         MOVZBL  #PRI$_TIMER,R2          ;SET PRIORITY INCREMENT CLASS
             FF15'    30          00E8   266         BSBW    SCH$QAST                ;QUEUE AST FOR PROCESS
                44    11          00EB   267         BRB     40$                     ;
    54    2C A5    3C          00ED   268 10$:    MOVZWL  TQE$L_RQPID(R5),R4      ;GET REQUESTING PROCESS INDEX
54  00000000'FF44    D0          00F1   269         MOVL    @L^SCH$GL_PCBVEC[R4],R4 ;GET PROCESS PCB ADDRESS
    2C A5    60 A4    D1          00F9   270         CMPL    PCB$L_PID(R4),TQE$L_RQPID(R5) ;PROCESS ID MATCH?
                0B    12          00FE   271         BNEQ    30$                     ;IF NEQ NO
             38 A4    B6          0100   272         INCW    PCB$W_ASTCNT(R4)        ;UPDATE AVAILABLE AST QUEUE ENTRIES
                06    11          0103   273         BRB     30$                     ;
          50    01    9A          0105   274 20$:    MOVZBL  #RSN$_ASTWAIT,R0        ;SET AST WAIT RESOURCE NUMBER
             FEF5'    30          0108   275         BSBW    SCH$RAVAIL              ;DECLARE RESOURCE AVAILABLE
          50    55    D0          010B   276 30$:    MOVL    R5,R0                   ;SET ADDRESS OF BLOCK TO DEALLOCATE
             FEEF'    30          010E   277         BSBW    EXE$DEANONPAGED         ;DEALLOCATE TIME QUEUE ENTRY
                1E    11          0111   278         BRB     40$                     ;
                                0113   279
                                0113   280 ;
                                0113   281 ; PROCESS WAKE UP
                                0113   282 ;
                                0113   283
    51    0C A5    D0          0113   284 WAKEUP: MOVL    TQE$L_PID(R5),R1        ;GET TARGET PROCESS ID
             FEE6'    30          0117   285         BSBW    SCH$WAKE                ;WAKE PROCESS
          D0 50    E9          011A   286         BLBC    R0,10$                  ;IF LBC PROCESS NOT IN SYSTEM
CB 0B A5    02    E1          011D   287         BBC     #TQE$V_REPEAT,TQE$B_RQTYPE(R5),10$ ;IF CLR, THEN NOT REPEATABLE
```

```
                50    20 A5    7D  0122    288 REPTIM: MOVQ    TQE$Q_DELTA(R5),R0      ;GET DELTA REPEAT TIME
                50    18 A5    C0  0126    289         ADDL    TQE$Q_TIME(R5),R0       ;ADD LOW ORDER PARTS OF TIME
                51    1C A5    D8  012A    290         ADWC    TQE$Q_TIME+4(R5),R1     ;ADD HIGH ORDER PARTS OF TIME
                      FECF'    30  012E    291         BSBW    EXE$INSTIMQ             ;INSERT ENTRY IN TIME QUEUE
                      FF44     31  0131    292 40$:    BRW     CHKTMQ                  ;
                               0134    293         .DSABL  LSB
```

```
                     0134   295          .SBTTL  SEARCH FOR TIME OUTS
                     0134   296 ;+
                     0134   297 ; EXE$TIMEOUT - SEARCH FOR TIME OUTS
                     0134   298 ;
                     0134   299 ; THIS ROUTINE IS ENTERED ONCE A SECOND TO PERFORM VARIOUS FUNCTIONS THAT
                     0134   300 ; NEED TO BE PERFORMED ONCE A SECOND.  THESE INCLUDE:
                     0134   301 ;
                     0134   302 ;       1)  SCAN THE DEVICE DATABASE FOR DEVICES THAT MAY HAVE TIMED OUT
                     0134   303 ;       2)  SCAN FOR CRB'S  THAT MAY HAVE TIMED OUT
                     0134   304 ;       3)  SCAN FOR WAITING LOCKS THAT MAY HAVE TIMED OUT (INITIATE
                     0134   305 ;           DEADLOCK SEARCH)
                     0134   306 ;       4)  WAKE THE SWAPPER, IF NECESSARY
                     0134   307 ;       5)  WAKE THE ERROR LOG PROCESS, IF NECESSARY
                     0134   308 ;       6)  SCAN FOR MEMORY CRD ERRORS AND REENABLE MEMORY INTERRUPTS
                     0134   309 ;       7)  UPDATE THE SYSTEM ABSOLUTE TIME IN SECONDS
                     0134   310 ;       8)  DECLARE A NON-PAGED DYNAMIC MEMORY AVAILABLE EVENT
                     0134   311 ;-
                     0134   312
                     0134   313 EXE$TIMEOUT::                            ;SEARCH FOR TIME OUTS
           56   DD   0134   314          PUSHL   R6                      ;SAVE A REGISTER
           55   DD   0136   315          PUSHL   R5                      ;SAVE ANOTHER
      56   63   D0   0138   316          MOVL    DDB$L_LINK(R3),R6       ;GET ADDRESS OF FIRST DDB
         FEC2'  30   013B   317          BSBW    SCH$SOPWAKE             ;WAKE SWAPPER IF NECESSARY
      0000'CF   D6   013E   318          INCL    W^EXE$GL_ABSTIM         ;UPDATE ABSOLUTE TIME IN SECONDS
03 0000'CF   01   E1   0142   319          BBC     #ERL$V_TIMER,W^ERL$GB_BUFFLAG,10$ ;IF CLR, TIMER NOT ACTIVE
         FEB5'  30   0148   320          BSBW    ERL$WAKE                ;WAKE ERROR LOG FORMAT PROCESS
         FEB2'  30   014B   321 10$:     BSBW    ECC$REENABLE            ;WAKE CRD INTERRUPT REENABLE AND SCAN
                     014E   322
                     014E   323 ;
                     014E   324 ; SCAN FOR DEVICE TIMEOUTS
                     014E   325 ;
                     014E   326
   55   04 A6   D0   014E   327 20$:     MOVL    DDB$L_UCB(R6),R5        ;GET ADDRESS OF FIRST UCB
           1A   13   0152   328          BEQL    60$                     ;EQL MEANS NO UCB'S AS YET ON THIS DDB
15 38 A5    14   E0   0154   329          BBS     #DEV$V_MBX,UCB$L_DEVCHAR(R5),60$ ;IF SET, DEVICE IS MAILBOX
3A 64 A5    00   E0   0159   330 30$:     BBS     #UCB$V_TIM,UCB$W_STS(R5),70$ ;IF SET, TIME OUT ENABLED
05 38 A5    02   E1   015E   331 40$:     BBC     #DEV$V_TRM,UCB$L_DEVCHAR(R5),50$ ;IF CLR, DEVICE NOT TERMINAL
6B 68 A5    01   E0   0163   332          BBS     #UCB$V_TT_TIMO,UCB$W_DEVSTS(R5),90$ ;IF SET, READ TIMEOUT ENABLED
   55   30 A5   D0   0168   333 50$:     MOVL    UCB$L_LINK(R5),R5       ;GET ADDRESS OF NEXT UCB
           EB   12   016C   334          BNEQ    30$                     ;IF NEQ MORE TO SCAN
      56   66   D0   016E   335 60$:     MOVL    DDB$L_LINK(R6),R6       ;GET ADDRESS OF NEXT DDB
           DB   12   0171   336          BNEQ    20$                     ;IF NEQ MORE TO SCAN
                     0173   337
                     0173   338 ;
                     0173   339 ; FINISHED DEVICE SCAN, NOW CHECK THE LIST OF CRB'S
                     0173   340 ;
                     0173   341
56   0000'CF   DE   0173   342          MOVAL   W^IOC$GL_CRBTMOUT,R6    ;PICK UP LIST HEAD
      56   66   D0   0178   343 65$:     MOVL    (R6),R6                 ;ANY MORE TO SCAN?
           7A   13   017B   344          BEQL    CHECK_FORK_N_WAIT       ;NO, DONE
                     017D   345          SETIPL  #IPL$_POWER             ;SET TO POWERFAIL
        04 A6   D1   0180   346          CMPL    CRB$L_DUETIME-CRB$L_TIMELINK(R6),-
      0000'CF        0183   347                  W^EXE$GL_ABSTIM         ;YES, IS THIS ONE DUE?
           0B   1A   0186   348          BGTRU   66$                     ;NO, SCAN AGAIN
   53   EC A6   DE   0188   349          MOVAL   -CRB$L_TIMELINK(R6),R3  ;YES, PICK UP POINTER TO CRB
   18 A3   01   CE   018C   350          MNEGL   #1, CRB$L_DUETIME(R3)   ;SET FOR NO MORE TIMEOUTS
        1C B3   16   0190   351          JSB     @CRB$L_TOUTROUT(R3)     ;CALL THE TIMEOUT ROUTINE
```

```
                                0193    352 66$:    SETIPL  #IPL$_TIMER             ;RESET THE IPL
                     E0   11    0196    353         BRB     65$                     ;CONTINUE SCAN
                                0198    354
                                0198    355
                                0198    356 ;
                                0198    357 ; DEVICE HAS ENABLED TIME OUT - SEE IF IT HAS TIMED OUT
                                0198    358 ;
                                0198    359
          0000'CF    6C A5  D1  0198    360 70$:    CMPL    UCB$L_DUETIM(R5),W^EXE$GL_ABSTIM ;POSSIBLE TIME OUT?
                        BE  1A  019E    361         BGTRU   40$                     ;IF GTRU NO
                                01A0    362         SETIPL  #IPL$_POWER             ;RAISE TO POWERFAIL IPL
            26 64 A5    00  E1  01A3    363         BBC     #UCB$V_TIM,UCB$W_STS(R5),80$ ;IF CLR, THEN TIME OUT NOT ENABLED
          0000'CF    6C A5  D1  01A8    364         CMPL    UCB$L_DUETIM(R5),W^EXE$GL_ABSTIM ;DEVICE TIME OUT?
                        1E  1A  01AE    365         BGTRU   80$                     ;IF GTRU NO
               64 A5    03  AA  01B0    366         BICW    #UCB$M_INT!UCB$M_TIM,UCB$W_STS(R5) ;DISABLE INTERRUPT AND TIMEOUT
          64 A5    0040 8F  A8  01B4    367         BISW    #UCB$M_TIMOUT,UCB$W_STS(R5) ;SET DEVICE TIMED OUT
                                01BA    368         SETIPL  UCB$B_DIPL(R5)          ;LOWER TO DEVICE IPL
               53    10 A5  7D  01BE    369         MOVQ    UCB$L_FR3(R5),R3        ;RETRIEVE SAVED R3 AND R4
               52    0C A5  D0  01C2    370         MOVL    UCB$L_FPC(R5),R2        ;GET SAVED PC
                  7E    72  32  01C6    371         CVTWL   -(R2),-(SP)             ;GET OFFSET TO EXCEPTION ROUTINE
                  52    8E  C0  01C9    372         ADDL    (SP)+,R2                ;CALCULATE ADDRESS OF EXCEPTION ROUTINE
                        62  16  01CC    373         JSB     (R2)                    ;CALL EXCEPTION ROUTINE
                                01CE    374 80$:    SETIPL  #IPL$_TIMER             ;LOWER IPL TO THAT OF TIMER
                        8B  11  01D1    375         BRB     40$                     ;
                                01D3    376
                                01D3    377 ;
                                01D3    378 ; TERMINAL READ TIMED IN PROGRESS
                                01D3    379 ;
                                01D3    380
          0000'CF    00B0 C5  D1  01D3    381 90$:    CMPL    UCB$L_TT_RDUE(R5),W^EXE$GL_ABSTIM ;TIME OUT POSSIBLE?
                        8C  1A  01DA    382         BGTRU   50$                     ;IF GTRU THEN NO
                                01DC    383         SETIPL  #IPL$_POWER             ;RAISE TO DEVICE IPL
             0D 68 A5    01  E1  01DF    384         BBC     #UCB$V_TT_TIMO,UCB$W_DEVSTS(R5),100$ ;IF CLR, TIMEOUT NOT ENABLED
          0000'CF    00B0 C5  D1  01E4    385         CMPL    UCB$L_TT_RDUE(R5),W^EXE$GL_ABSTIM ;TIMED OUT?
                        04  1A  01EB    386         BGTRU   100$                    ;IF GTRU THEN NO
                  00B4 D5  16  01ED    387         JSB     @UCB$L_TT_RTIMOU(R5)    ;GO OFF TO THE TERMINAL SERVICE
                                01F1    388 100$:   SETIPL  #IPL$_TIMER             ;LOWER IPL TO THAT OF TIMER
                     FF71   31  01F4    389         BRW     50$                     ;
                                01F7    390
                                01F7    391
                                01F7    392 ;
                                01F7    393 ; PROCESS THE FORK-AND-WAIT WORK QUEUE
                                01F7    394 ;
                                01F7    395 ; To avoid an infinite loop of executing new entries placed on the work queue
                                01F7    396 ; by fork threads resumed from the work queue, the entire work queue is
                                01F7    397 ; removed from the normal header and hung on a header allocated on the stack.
                                01F7    398 ; The normal queue header is then initialized thus providing a proper target
                                01F7    399 ; of new queue insertion operations.
                                01F7    400 ;
                                01F7    401         ASSUME  FKB$L_FR4 EQ <FKB$L_FR3 + 4>
                                01F7    402
                                01F7    403 CHECK_FORK_N_WAIT:
               56    0000'CF    DE  01F7    404         MOVAL   W^EXE$GL_FKWAITFL, R6   ; Get fork-&-wait queue header address.
                  56    66    D1  01FC    405         CMPL    (R6), R6                ; Is the queue empty?
                        2D    13  01FF    406         BEQL    CHECK_LOCKS             ; Branch if queue is empty.
                  5E    08    C2  0201    407         SUBL    #8, SP                  ; Make space on stack for header.
                  55    66    0F  0204    408         REMQUE  (R6), R5                ; Dequeue the work queue header.
```

```
                04 B5    6E    0E  0207   409          INSQUE  (SP), @4(R5)            ; Queue stack header in its place.
                   66    56    D0  020B   410          MOVL    R6, (R6)                ; Make real fork=&-wait queue empty.
                04 A6    56    D0  020E   411          MOVL    R6, 4(R6)               ;
                55   00  BE    0F  0212   412 10$:     REMQUE  @(SP), R5               ; Get an entry from stack work queue.
                         13    1D  0216   413          BVS     90$                     ; Branch if stack work queue empty.
                53   10  A5    7D  0218   414          MOVQ    FKB$L_FR3(R5), R3       ; Restore fork context.
                                  021C   415          DSBINT  FKB$B_FIPL(R5)          ; Establish fork IPL.
                     0C  B5    16  0223   416          JSB     @FKB$L_FPC(R5)          ; Restart fork thread.
                                  0226   417          ENBINT                          ; Restore our IPL.
                         E7    11  0229   418          BRB     10$                     ; Loop through entire stack work queue.
                    5E   08    C0  022B   419 90$:     ADDL    #8, SP                  ; All done: pop queue header from stack.
                                  022E   420 ;        JMP     CHECK_LOCKS             ; Then, fall through to checking locks.
                                  022E   421
                                  022E   422
                                  022E   423 ;
                                  022E   424 ; SCAN FOR WAITING LOCKS THAT MAY HAVE TIMED OUT.  INITIATE A DEADLOCK
                                  022E   425 ; SEARCH IF ONE IS FOUND.
                                  022E   426 ;
                                  022E   427
                                  022E   428 CHECK_LOCKS:
             55   0000'CF    DE  022E   429          MOVAL   W^LCK$GL_TIMOUTQ,R5     ;GET ADDRESS OF LIST HEAD
                  56   65    D0  0233   430 10$:     MOVL    (R5),R6                 ;GET FIRST ENTRY ON LIST
                  56   55    D1  0236   431          CMPL    R5,R6                   ;IS LIST EMPTY?
                         0B    13  0239   432          BEQL    20$                     ;YES
                    18   A6    D1  023B   433          CMPL    LKB$L_DUETIME(R6),-     ;NO, HAS THIS ONE TIMED OUT?
                  0000'CF                 023E   434                  W^EXE$GL_ABSTIM
                         03    1A  0241   435          BGTRU   20$                     ;NO, ALSO NO NEED TO LOOK FURTHER
                                  0243   436                                          ;AS LIST IS ORDERED
                     FDBA'      30  0243   437          BSBW    LCK$SEARCHDLCK          ;SEARCH FOR DEADLOCK
                                  0246   438                                          ;DON'T REPEAT LOOP HERE; IT'S DONE
                                  0246   439                                          ;IN DEADLOCK
                                  0246   440 20$:
                                  0246   441 ;
                                  0246   442 ; SCAN PCB VECTOR FOR PROCESSES NEEDING PRIORITY BOOST
                                  0246   443 ;
                                  0246   444 SCAN_PROC:
                       55   0000'CF    3C  0246   445          MOVZWL  W^SGN$GW_PIXSCAN,R5     ;GET NUMBER OF PROCESSES TO SCAN
                                  52    13  024B   446          BEQL    100$                    ;NONE, DO NOTHING
                       56   0000'CF    3C  024D   447          MOVZWL  W^EXE$GW_SCANPIX,R6     ;GET PIX TO CHECK
                  54   00000000'FF46    D0  0252   448 10$:     MOVL    @L^SCH$GL_PCBVEC[R6],R4 ;FETCH PCB ADDRESS FOR PIX
                         60   A4    56    B1  025A   449          CMPW    R6,PCB$L_PID(R4)        ;IS THIS AN ACTIVE PCB?
                                  2C    12  025E   450          BNEQ    20$                     ;NO, SKIP IT.
                     50   2C   A4    01    A9  0260   451          BISW3   #1,PCB$W_STATE(R4),R0    ;GET STATE = 1
                                  50    0D    B1  0265   452          CMPW    #<SCH$C_COM!1>,R0       ;IS STATE COM OR COMO?
                                  22    12  0268   453          BNEQ    20$                     ;BR IF NOT
                       50   00000000'EF    D0  026A   454          MOVL    L^SCH$GL_COMQS,R0       ;GET SUMMARY LONGWORD
             51   00000000'EF    05    00    EF  0271   455          EXTZV   #0,#5,L^SCH$GB_PRI,R1   ;GET CURRENT PRIORITY
                         00 50    51    E2  027A   456          BBSS    R1,R0,15$               ;SET BIT FOR CURRENT PROCESS
                  50   50   0F    10    EA  027E   457 15$:     FFS     #16,#15,R0,R0           ;GET HIGHEST BACKGROUND PRIORITY
                         0B A4    50    91  0283   458          CMPB    R0,PCB$B_PRI(R4)        ;IS PROCESS ALREADY HIGHER PRIORITY?
                                  03    18  0287   459          BGEQ    20$                     ;BR IF SO
                                FD74'   30  0289   460          BSBW    SCH$CHSEP               ;SET TEMPORARY PRIORITY BOOST
                  03 56   00000000'EF    F3  028C   461 20$:     AOBLEQ  L^SCH$GL_MAXPIX,R6,30$  ;NEXT PROCESS INDEX
                                  56    02    D0  0294   462          MOVL    #2,R6                   ;SET TO START PAST SWAPPER AND NULL
                                    B8  55    F5  0297   463 30$:     SOBGTR  R5,10$                  ;SCAN FOR NUMBER OF PROCESSES REQUESTED
                         0000'CF    56    B0  029A   464          MOVW    R6,W^EXE$GW_SCANPIX     ;SAVE NEW VALUE
                                  029F   465 100$:                                ;
```

```
                     029F    466 ;
                     029F    467 ; DECLARE NON-PAGED AND PAGED DYNAMIC MEMORY AVAILABLE.  THIS IS NECESSARY
                     029F    468 ; BECAUSE IN CERTAIN RARE CASES, THE MEMORY ALLOCATION ROUTINES MAY FAIL TO DO
                     029F    469 ; THIS AS OFTEN AS NECESSARY.
                     029F    470 ; THE MAILBOX RESOURCE IS ALSO DECLARED AVAILABLE FOR SIMILAR REASONS.
                     029F    471 ;
                     029F    472
50   03    3C      029F    473          MOVZWL  #RSN$_NPDYNMEM,R0       ;NON-PAGED DYNAMIC MEMORY RESOURCE
   FD5B'   30      02A2    474          BSBW    SCH$RAVAIL              ;DECLARE RESOURCE AVAILABLE
50   05    3C      02A5    475          MOVZWL  #RSN$_PGDYNMEM,R0       ;PAGED DYNAMIC MEMORY RESOURCE
   FD55'   30      02A8    476          BSBW    SCH$RAVAIL              ;DECLARE RESOURCE AVAILABLE
50   02    3C      02AB    477          MOVZWL  #RSN$_MAILBOX,R0        ;MAILBOX RESOURCE
   FD4F'   30      02AE    478          BSBW    SCH$RAVAIL              ;DECLARE RESOURCE AVAILABLE
                     02B1    479
                     02B1    480 ;
                     02B1    481 ; ALL DONE - RETURN
                     02B1    482 ;
                     02B1    483
                     02B1    484 TIMEOUT_DONE:
55   8E    7D      02B1    485          MOVQ    (SP)+,R5                ;RESTORE REGISTERS
         05      02B4    486          RSB                             ;RETURN
                     02B5    487
                     02B5    488          .END
```

```
ACB$V_QUOTA                     = 00000006
BUG$_INVTQEFMT                    ********    X   02
CA$_MEASURE                     = 00000002
CHECK_FORK_N_WAIT                 000001F7 R      02
CHECK_LOCKS                       0000022E R      02
CHKTMQ                            00000078 R      02
CRB$L_DUETIME                   = 00000018
CRB$L_TIMELINK                  = 00000014
CRB$L_TOUTROUT                  = 0000001C
DDB$L_LINK                      = 00000000
DDB$L_UCB                       = 00000004
DEV$V_MBX                       = 00000014
DEV$V_TRM                       = 00000002
ECC$REENABLE                      ********    X   02
ERL$GB_BUFFLAG                    ********    X   02
ERL$V_TIMER                     = 00000001
ERL$WAKE                          ********    X   02
EXE$DEANONPAGED                   ********    X   02
EXE$GL_ABSTIM                     ********    X   02
EXE$GL_FKWAITFL                   ********    X   02
EXE$GL_TQFL                       ********    X   02
EXE$GQ_SYSTIME                    ********    X   02
EXE$GW_SCANPIX                    ********    X   02
EXE$HWCLKINT                      00000000 RG     02
EXE$INSTIMQ                       ********    X   02
EXE$SWTIMINT                      00000060 RG     02
EXE$TIMEOUT                       00000134 RG     02
EXE$UBCLKINT                      00000008 RG     02
FKB$B_FIPL                      = 0000000B
FKB$L_FPC                       = 0000000C
FKB$L_FR3                       = 00000010
FKB$L_FR4                       = 00000014
IOC$GL_CRBTMOUT                   ********    X   02
IPL$_HWCLK                      = 00000018
IPL$_POWER                      = 0000001F
IPL$_SYNCH                      = 00000008
IPL$_TIMER                      = 00000008
IPL$_TIMERFORK                  = 00000007
JIB$W_TQCNT                     = 00000034
LCK$GL_TIMOUTQ                    ********    X   02
LCK$SEARCHDLCK                    ********    X   02
LKB$L_DUETIME                   = 00000018
PCB$B_PRI                       = 0000000B
PCB$L_JIB                       = 00000080
PCB$L_PHD                       = 0000006C
PCB$L_PID                       = 00000060
PCB$W_ASTCNT                    = 00000038
PCB$W_STATE                     = 0000002C
PHD$L_CPUTIM                    = 00000038
PHD$W_QUANT                     = 0000003C
  MS$GL_KERNEL                    ********    X   02
PR$_ICCS                        = 00000018
PR$_IPL                         = 00000012
PR$_SIRR                        = 00000014
PRI$_TIMER                      = 00000002
PSL$V_IS                        = 0000001A
REPTIM                            00000122 R      02
RSN$_ASTWAIT                    = 00000001
RSN$_MAILBOX                    = 00000002
RSN$_NPDYNMEM                   = 00000003
RSN$_PGDYNMEM                   = 00000005
SCAN_PROC                         00000246 R      02
SCH$CHSEP                         ********    X   02
SCH$C_COM                       = 0000000C
SCH$GB_PRI                        ********    X   02
SCH$GL_COMQS                      ********    X   02
SCH$GL_CURPCB                     ********    X   02
SCH$GL_MAXPIX                     ********    X   02
SCH$GL_PCBVEC                     ********    X   02
SCH$POSTEF                        ********    X   02
SCH$QAST                          ********    X   02
SCH$QEND                          ********    X   02
SCH$RAVAIL                        ********    X   02
SCH$SWPWAKE                       ********    X   02
SCH$WAKE                          ********    X   02
SGN$GW_PIXSCAN                    ********    X   02
SYSUB                             000000B4 R      02
TIMEOUT_DONE                      000002B1 R      02
TIMER                             000000C2 R      02
TQE$B_EFN                       = 00000029
TQE$B_RMOD                      = 00000028
TQE$B_RQTYPE                    = 0000000B
TQE$L_FPC                       = 0000000C
TQE$L_FR3                       = 00000010
TQE$L_PID                       = 0000000C
TQE$L_RQPID                     = 0000002C
TQE$Q_DELTA                     = 00000020
TQE$Q_TIME                      = 00000018
TQE$V_REPEAT                    = 00000002
UCB$B_DIPL                      = 0000005E
UCB$L_DEVCHAR                   = 00000038
UCB$L_DUETIM                    = 0000006C
UCB$L_FPC                       = 0000000C
UCB$L_FR3                       = 00000010
UCB$L_LINK                      = 00000030
UCB$L_TT_RDUE                   = 000000B0
UCB$L_TT_RTIMOU                 = 000000B4
UCB$M_INT                       = 00000002
UCB$M_TIM                       = 00000001
UCB$M_TIMOUT                    = 00000040
UCB$V_TIM                       = 00000000
UCB$V_TT_TIMO                   = 00000001
UCB$W_DEVSTS                    = 00000068
UCB$W_STS                       = 00000064
WAKEUP                            00000113 R      02
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
A$EXENONPAGED	000002B5 (693.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	38	00:00:00.04	00:00:01.52
Command processing	128	00:00:00.54	00:00:05.4
Pass 1	438	00:00:16.90	00:00:56.05
Symbol table sort	0	00:00:02.74	00:00:08.30
Pass 2	106	00:00:02.98	00:00:11.11
Symbol table output	12	00:00:00.10	00:00:00.10
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	726	00:00:23.33	00:01:22.51

The working set limit was 1650 pages.
92546 bytes (181 pages) of virtual memory were used to buffer the intermediate code.
There were 100 pages of symbol table space allocated to hold 1719 non-local and 33 local symbols.
488 source lines were read in Pass 1, producing 17 object records in Pass 2.
44 pages of virtual memory were used to define 43 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	32
_255DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	39

1909 GETS were required to define 39 macros.

There were no errors, warnings or information messages.

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

WRTMFYPAG
LIS

UCBCREDEL
LIS

USRVECTOR
LIS

SYSVECTOR
LIS

SYSINIT
LIS

VERSION
LIS

SYSINI

UPCASEDAT
LIS

SYSINIT
MAP

SYSWAIT
LIS

TIMESCHDL
LIS