```
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

```
LL              IIIIII    BBBBBBBB    EEEEEEEEEE  FFFFFFFFFF
LL              IIIIII    BBBBBBBB    EEEEEEEEEE  FFFFFFFFFF
LL                II      BB      BB  EE          FF
LL                II      BB      BB  EE          FF
LL                II      BB      BB  EE          FF
LL                II      BB      BB  EE          FF
LL                II      BBBBBBBB    EEEEEEEE    FFFFFFFF
LL                II      BBBBBBBB    EEEEEEEE    FFFFFFFF
LL                II      BB      BB  EE          FF
LL                II      BB      BB  EE          FF
LL                II      BB      BB  EE          FF              ....
LL                II      BB      BB  EE          FF              ....
LLLLLLLLLL      IIIIII    BBBBBBBB    EEEEEEEEEE  FF              ....
LLLLLLLLLL      IIIIII    BBBBBBBB    EEEEEEEEEE  FF              ....


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
```

```
0000     1         .TITLE  LIB$EF - Resource allocator for local event flags
0000     2         .IDENT  /1-006/                 ; File: LIBEF.MAR  Edit: MDL1006
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28
0000    29 ;++
0000    30 ; FACILITY: General Utility Library
0000    31 ;
0000    32 ; ABSTRACT:
0000    33 ;
0000    34 ;       Three routines for allocating and deallocating local event
0000    35 ;       flag numbers.  Using these routines allows use of local
0000    36 ;       event flags by multiple procedures without conflicts.
0000    37 ;
0000    38 ; ENVIRONMENT: User Mode, AST Reentrant
0000    39 ;
0000    40 ;--
0000    41 ; AUTHOR: Steven B. Lionel, CREATION DATE: 08-DEC-78
0000    42 ;
0000    43 ; MODIFIED BY:
0000    44 ;
0000    45 ; SBL, : VERSION 1
0000    46 ; 1-001 - Original
0000    47 ; 1-002 - Pre-reserve event flags 1-23.  SBL 18-Dec-78
0000    48 ; 1-003 - Put   before PSECT names and make error codes be
0000    49 ;          global.  JBS for SBL 23-JAN-1979
0000    50 ; 1-004 - Correct a typo in edit 003.  JBS 24-JAN-1979
0000    51 ; 1-005 - Add a first time flag to allow initialization of non-zero data at
0000    52 ;         run-time, thus allowing the linker to perform demand-zero compression
0000    53 ;         and helping performance.  MDL 6-Jul-1984
0000    54 ; 1-006 - Change interpretation of 1/2 the bits in EF_POOL, eliminating the need for
0000    55 ;         a first time flag and restoring AST re-entrancy.  MDL 7-Aug-1984
```

```
                     0000     57          .SBTTL  DECLARATIONS
                     0000     58 ;
                     0000     59 ; INCLUDE FILES:
                     0000     60 ;
                     0000     61
                     0000     62 ;
                     0000     63 ; EXTERNAL DECLARATIONS:
                     0000     64 ;
                     0000     65          .DSABL  GBL                             ; Prevent undeclared
                     0000     66                                                  ; symbols from being
                     0000     67                                                  ; automatically global.
                     0000     68          .EXTRN  LIB$_INSEF                      ; Error codes
                     0000     69          .EXTRN  LIB$_EF_ALRRES
                     0000     70          .EXTRN  LIB$_EF_ALRFRE
                     0000     71          .EXTRN  LIB$_EF_RESSYS
                     0000     72
                     0000     73 ;
                     0000     74 ; MACROS:
                     0000     75 ;
                     0000     76
                     0000     77 ;
                     0000     78 ; EQUATED SYMBOLS:
                     0000     79 ;
 00000004            0000     80          ef_number = 4               ; event flag number parameter
                     0000     81 ;
                     0000     82 ; OWN STORAGE:
                     0000     83 ;
 00000000            0000     84          .PSECT  _LIB$DATA RD, WRT, NOEXE, NOSHR, PIC, LONG, -
                     0000     85                          CON, LCL, REL, USR
                     0000     86
                     0000     87 EF_POOL:                                ; Pool of available event flags
 00000000 00000000   0000     88          .QUAD   0
                     0008     89
                     0008     90          ; P L E A S E   N O T E   ! ! !
                     0008     91          ;
                     0008     92          ; EF_POOL is structured as follows:
                     0008     93          ; Low order bit (bit 0) represents event flag number 63.
                     0008     94          ; High order bit (bit 63) represents event flag number 0.
                     0008     95          ;
                     0008     96          ; Bits 0-31 of EF_POOL (event flags 32-63) are interpreted as folows:
                     0008     97          ; 0 => available
                     0008     98          ; 1 => reserved
                     0008     99          ; Bits 32-63 of EF_POOL (event flags 0-31) are interpreted as follows:
                     0008    100          ; 0 => reserved
                     0008    101          ; 1 => available
                     0008    102          ;
                     0008    103          ; Thus, the initial setting of 0 allows for:
                     0008    104          ; Event flags 0 and  24-31 are reserved to the system.
                     0008    105          ; Event flags 1-23 are pre-reserved.
                     0008    106          ; Event flags 32-63 are available for use.
                     0008    107
                     0008    108 ;
                     0008    109 ; PSECT DECLARATIONS:
                     0008    110 ;
                     0008    111          .PSECT _LIB$CODE PIC, USR, CON, REL, LCL, SHR, -
 00000000            0000    112                          EXE, RD, NOWRT, LONG
                     0000    113
```

```
                                        .SBTTL  LIB$GET_EF - Allocate one local event flag
                                ;++
                                ; FUNCTIONAL DESCRIPTION:
                                ;
                                ;       LIB$GET_EF allocates one local event flag from a process-wide
                                ;       pool.  If a flag is available for use, its number is returned
                                ;       to the caller.  If no flags are available, an error is returned
                                ;       as the function value.
                                ;
                                ; CALLING SEQUENCE:
                                ;
                                ;       status.wlc.v = LIB$GET_EF (ef_number.wl.r)
                                ;
                                ; INPUT PARAMETERS:
                                ;
                                ;       NONE
                                ;
                                ; IMPLICIT INPUTS:
                                ;
                                ;       EF_POOL, a table of available event flag numbers in OWN
                                ;       storage.
                                ;
                                ; OUTPUT PARAMETERS:
                                ;
                                ;       ef_number       - The number of the local event flag allocated
                                ;                         or -1 if none were available.
                                ;
                                ; IMPLICIT OUTPUTS:
                                ;
                                ;       If successful, an entry is made into EF_POOL indicating that
                                ;       a local event flag has been reserved.
                                ;
                                ; FUNCTION VALUE:
                                ; COMPLETION CODES:
                                ;
                                ;       SS$_NORMAL      - Routine successfully completed.
                                ;
                                ;       LIB$_INSEF      - Insufficient event flags.  There were no
                                ;                         more event flags available for allocation.
                                ;                         If this error is returned, ef_number is
                                ;                         also set to -1 in case the caller does not
                                ;                         check for failure.
                                ;
                                ; SIDE EFFECTS:
                                ;
                                ;       NONE
                                ;
                                ;--
                                
                        4000    .ENTRY LIB$GET_EF, ^M<IV>       ; Save nothing
                                
                                ;+
                                ;       Scan EF_POOL for first available event flag number
                                ;-
                                
                                SCAN:
50  00000000'EF   20    00  EB     FFC     #0, #32, EF_POOL, R0
```

```
                     15  12  000B  172          BNEQ    FOUND                       ; One was found
    50  00000000'EF 20  20  EA  000D  173          FFS     #32, #32, EF_POOL, R0       ; Try next 32 positions
                     1B  13  0016  174          BEQL    ALL_OUT                     ; None available
                             0018  175
                             0018  176 ;+
                             0018  177 ;       Now recheck and set the bit under interlock, in case someone
                             0018  178 ;       has set it at AST level in the meantime.
                             0018  179 ;-
                             0018  180
                             0018  181 FOUND_LOW:
        E2 00000000'EF       50  E5  0018  182          BBCC    R0, EF_POOL, SCAN       ; Repeat scan if already set
                             08  11  0020  183          BRB     LEAVE_GET
                             0022  184
                             0022  185 FOUND:
        D8 00000000'EF       50  E2  0022  186          BBSS    R0, EF_POOL, SCAN       ; Repeat scan if already set
                             002A  187
                             002A  188 ;+
                             002A  189 ;       Return success with event flag number in ef_number.
                             002A  190 ;-
                             002A  191
                             002A  192 LEAVE_GET:
            04 BC    3F      50  C3  002A  193          SUBL3   R0, #63, @ef_number(AP) ; Subtract from 63 because
                             002F  194                                               ; lo order table bit is
                             002F  195                                               ; event flag 63.
                             002F  196
                     50  01  D0  002F  197          MOVL    #1, R0                  ; SS$_NORMAL
                             04  0032  198          RET                             ; Exit
                             0033  199
                             0033  200 ;+
                             0033  201 ;       Return error since no event flags available
                             0033  202 ;-
                             0033  203
                             0033  204 ALL_OUT:
            04 BC    01      CE  0033  205          MNEGL   #1, @ef_number(AP)      ; Set ef-number to -1
    50  00000000'8F          D0  0037  206          MOVL    #LIB$_INSEF, R0         ; Insufficient event flags
                             04  003E  207          RET                             ; Exit
                             003F  208
```

```
                     ;  LIB$FREE_EF - Deallocate one local event flag
                     ;++
                     ; FUNCTIONAL DESCRIPTION:
                     ;
                     ;       LIB$FREE_EF is the complement of LIB$GET_EF.  When a routine
                     ;       called LIB$GET_EF to allocate a local event flag, and no
                     ;       longer needs it, LIB$FREE_EF should be called to free the
                     ;       event flag for use by other routines.
                     ;
                     ; CALLING SEQUENCE:
                     ;
                     ;       status.wlc.v = LIB$FREE_EF (ef_number.rl.r)
                     ;
                     ; INPUT PARAMETERS:
                     ;
                     ;       ef_number       - The number of the event flag to be
                     ;                         deallocated.  This is the value returned
                     ;                         to the user by LIB$GET_EF.
                     ;
                     ; IMPLICIT INPUTS:
                     ;
                     ;       EF_POOL, a table of available event flag numbers in OWN
                     ;       storage.
                     ;
                     ; OUTPUT PARAMETERS:
                     ;
                     ;       NONE
                     ;
                     ; IMPLICIT OUTPUTS:
                     ;
                     ;       An entry is made in EF_POOL indicating that the event flag
                     ;       is free for use.
                     ;
                     ; FUNCTION VALUE:
                     ; COMPLETION CODES:
                     ;
                     ;       SS$_NORMAL      - Routine successfully completed.
                     ;
                     ;       LIB$_EF_ALRFRE  - Event flag already free.
                     ;
                     ;       LIB$_EF_RESSYS  - Event flag reserved to system.  This
                     ;                         occurs if ef_number is outside the ranges
                     ;                         of 1-23 and 32-63.
                     ;
                     ; SIDE EFFECTS:
                     ;
                     ;       NONE
                     ;
                     ;--
                     
                     .ENTRY LIB$FREE_EF, ^M<IV>      ; Save nothing
                     
                     ;+
                     ;       Check to see if ef_number is in the proper range.
                     ;-
                     
3F 04 BC D1          CMPL    @ef_number(AP), #63     ; Bigger than 63?
```

```
                     31   14  0045   267          BGTR    RES_SYS_1                       ; Yes, error
                  04 BC   D5  0047   268          TSTL    @ef_number(AP)                  ; Less than 1?
                     2C   15  004A   269          BLEQ    RES_SYS_1                       ; Yes, error
               17    04 BC   91  004C   270          CMPB    @ef_number(AP), #23             ; 1-23?
                     15   15  0050   271          BLEQ    OK_LOW                          ; Yes, ok
               20    04 BC   91  0052   272          CMPB    @ef_number(AP), #32             ; 32-63?
                     20   19  0056   273          BLSS    RES_SYS_1                       ; No, error
                              0058   274
                              0058   275 ;+
                              0058   276 ;       ef_number is in range 32-63.  Now, unset the bit.
                              0058   277 ;-
                              0058   278
                              0058   279 OK_1:
          50   3F   04 BC   C3  0058   280          SUBL3   @ef_number(AP), #63, R0 ; Convert to bit offset
    1B 00000000'EF    50   E5  005D   281          BBCC    R0, EF_POOL, ALR_FRE    ; Clear but error if
                              0065   282                                          ; already clear.
                     0D   11  0065   283          BRB     LEAVE_FRE
                              0067   284
                              0067   285 ;+
                              0067   286 ;       ef_number is in range 1-23.  Now, set the bit.
                              0067   287 ;-
                              0067   288
                              0067   289 OK_LOW:
          50   3F   04 BC   C3  0067   290          SUBL3   @ef_number(AP), #63, R0 ; Convert to bit offset
    0C 00000000'EF    50   E2  006C   291          BBSS    R0, EF_POOL, ALR_FRE    ; SET but error if
                              0074   292                                          ; already SET.
                              0074   293
                              0074   294 ;+
                              0074   295 ;       Return success
                              0074   296 ;-
                              0074   297
                              0074   298 LEAVE_FRE:
               50   01   D0  0074   299          MOVL    #1, R0                          ; SS$_NORMAL
                     04   04  0077   300          RET
                              0078   301
                              0078   302 ;+
                              0078   303 ;       Error if event flag number reserved to system or
                              0078   304 ;       out of range.
                              0078   305 ;-
                              0078   306
                              0078   307 RES_SYS_1:
    50    00000000'8F   D0  0078   308          MOVL    #LIB$_EF_RESSYS, R0     ; Event flag reserved
                     04   04  007F   309          RET
                              0080   310
                              0080   311 ;+
                              0080   312 ;       Error if event flag already free.
                              0080   313 ;-
                              0080   314
                              0080   315 ALR_FRE:
    50    00000000'8F   D0  0080   316          MOVL    #LIB$_EF_ALRFRE, R0     ; Event flag already free
                     04   04  0087   317          RET
                              0088   318
```

```
                        .SBTTL  LIB$RESERVE_EF - Reserve a local event flag
                ;++
                ; FUNCTIONAL DESCRIPTION:
                ;
                ;       LIB$RESERVE_EF is used when a routine wants to allocate
                ;       a particular local event flag number.  This is different
                ;       from LIB$GET_EF which allocates an arbitrary event flag.
                ;
                ;       To deallocate an event flag reserved with LIB$RESERVE_EF
                ;       use LIB$FREE_EF.
                ;
                ; CALLING SEQUENCE:
                ;
                ;       status.wlc.v = LIB$RESERVE_EF (ef_number.rl.r)
                ;
                ; INPUT PARAMETERS:
                ;
                ;       ef_number       - The number of the event flag desired to be
                ;                         allocated.
                ;
                ; IMPLICIT INPUTS:
                ;
                ;       EF_POOL, a table of available event flags located in OWN
                ;       storage.
                ;
                ; OUTPUT PARAMETERS:
                ;
                ;       NONE
                ;
                ; IMPLICIT OUTPUTS:
                ;
                ;       An entry is made in EF_POOL indicating that the event
                ;       flag is allocated.
                ;
                ; FUNCTION VALUE:
                ; COMPLETION CODES:
                ;
                ;       SS$_NORMAL      - Routine successfully completed.
                ;
                ;       LIB$_EF_ALRRES  - Event flag already reserved.
                ;
                ;       LIB$_EF_RESSYS  - Event flag reserved to system.  This
                ;                         occurs if ef_number is outside the ranges
                ;                         of 1-23 and 32-63.
                ;
                ; SIDE EFFECTS:
                ;
                ;       NONE
                ;
                ;--
                
4000            .ENTRY LIB$RESERVE_EF, ^M<IV>   ; Save nothing
                
                ;+
                ;       First check to see if ef_number is in range.
                ;-
```

```
                3F   04 BC   D1  008A   377          CMPL    @ef_number(AP), #63     ; Greater than 63?
                          31   14  008E   378          BGTR    RES_SYS_2               ; Yes, error
                     04 BC   D5  0090   379          TSTL    @ef_number(AP)          ; Less than 1?
                          2C   15  0093   380          BLEQ    RES_SYS_2               ; Yes, error
                17   04 BC   91  0095   381          CMPB    @ef_number(AP), #23     ; 1-23?
                          15   15  0099   382          BLEQ    OK_LOW2                 ; Yes, ok
                20   04 BC   91  009B   383          CMPB    @ef_number(AP), #32     ; 32-63?
                          20   19  009F   384          BLSS    RES_SYS_2               ; No, error
                                   00A1   385
                                   00A1   386 ;+
                                   00A1   387 ;       Now attempt to allocate an event flag in the 32-63 range.
                                   00A1   388 ;-
                                   00A1   389
                                   00A1   390 OK_2:
             50    3F   04 BC   C3  00A1   391          SUBL3   @ef_number(AP), #63, R0 ; Convert to bit offset
1B 00000000'EF     50   E2  00A6   392          BBSS    R0, EF_POOL, ALR_RES    ; Reserve it, but error
                                   00AE   393                                  ; if already reserved.
                          0D   11  00AE   394          BRB     LEAVE_RES
                                   00B0   395
                                   00B0   396 ;+
                                   00B0   397 ;       Now attempt to allocate an event flag in the 1-23 range.
                                   00B0   398 ;-
                                   00B0   399
                                   00B0   400 OK_LOW2:
             50    3F   04 BC   C3  00B0   401          SUBL3   @ef_number(AP), #63, R0 ; Convert to bit offset
0C 00000000'EF     50   E5  00B5   402          BBCC    R0, EF_POOL, ALR_RES    ; Reserve it, but error
                                   00BD   403                                  ; if already reserved.
                                   00BD   404
                                   00BD   405 ;+
                                   00BD   406 ;       Return success
                                   00BD   407 ;-
                                   00BD   408
                                   00BD   409 LEAVE_RES:
                50   01   D0  00BD   410          MOVL    #1, R0                  ; SS$_NORMAL
                          04   00C0   411          RET                             ; Exit
                                   00C1   412
                                   00C1   413 ;+
                                   00C1   414 ;       Error if event flag number reserved to system or out
                                   00C1   415 ;       of range.
                                   00C1   416 ;-
                                   00C1   417
                                   00C1   418 RES_SYS_2:
50   00000000'8F   D0  00C1   419          MOVL    #LIB$_EF_RESSYS, R0     ; Reserved to system
                          04   00C8   420          RET                             ; Exit
                                   00C9   421
                                   00C9   422 ;+
                                   00C9   423 ;       Error if already allocated.
                                   00C9   424 ;-
                                   00C9   425
                                   00C9   426 ALR_RES:
50   00000000'8F   D0  00C9   427          MOVL    #LIB$_EF_ALRRES, R0     ; Already reserved
                          04   00D0   428          RET                             ; Exit
                                   00D1   429
                                   00D1   430          .END
```

```
ALL_OUT             00000033 R      02
ALR_FRE             00000080 R      02
ALR_RES             000000C9 R      02
EF_NUMBER       =   00000004
EF_POOL             00000000 R      01
FOUND               00000022 R      02
FOUND_LOW           00000018 R      02
LEAVE_FRE           00000074 R      02
LEAVE_GET           0000002A R      02
LEAVE_RES           000000BD R      02
LIB$FREE_EF         0000003F RG     02
LIB$GET_EF          00000000 RG     02
LIB$RESERVE_EF      00000088 RG     02
LIB$_EF_ALRFRE      ********   X    00
LIB$_EF_ALRRES      ********   X    00
LIB$_EF_RESSYS      ********   X    00
LIB$_INSEF          ********   X    00
OK_1                00000058 R      02
OK_2                000000A1 R      02
OK_LOW              00000067 R      02
OK_LOW2             000000B0 R      02
RES_SYS_1           00000078 R      02
RES_SYS_2           000000C1 R      02
SCAN                00000002 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
_LIB$DATA	00000008 (8.)	01 (1.)	PIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG
_LIB$CODE	000000D1 (209.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.03	00:00:02.00
Command processing	125	00:00:00.32	00:00:02.10
Pass 1	73	00:00:00.52	00:00:03.20
Symbol table sort	0	00:00:00.00	00:00:00.00
Pass 2	81	00:00:00.49	00:00:03.37
Symbol table output	3	00:00:00.04	00:00:00.47
Psect synopsis output	3	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	319	00:00:01.42	00:00:11.16

The working set limit was 1050 pages.
4949 bytes (10 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 24 non-local and 0 local symbols.
430 source lines were read in Pass 1, producing 19 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.

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

Macro library name                 Macros defined
------------------                 --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2              0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:LIBEF/OBJ=OBJ$:LIBEF MSRC$:LIBEF/UPDATE=(ENH$:LIBEF)
```

LIBEMODH
LIS

LIBEMODD
LIS

LIBESTEMU
LIS

LIBEF
LIS

LIBEMULAT
LIS

LIBFFS
LIS

LIBFINCVT
LIS

LIBE2AREV
LIS

LIBFAO
LIS

LIBEMODG
LIS

LIBEXTV
LIS

LIBEDIV
LIS

LIBESTABL
LIS

LIBFFC
LIS

LIBEMODF
LIS

LIBEMUL
LIS

LIBFILSCA
LIS

LIBEXTZV
LIS

LIBEBCASC
LIS

LIBFAOL
LIS