```
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS     CCCCCCCCCCC     AAAAAAAAA     LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS     CCCCCCCCCCC     AAAAAAAAA     LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS     CCCCCCCCCCC     AAAAAAAAA     LLL
PPP         PPP    AAA       AAA    SSS             CCC             AAA       AAA   LLL
PPP         PPP    AAA       AAA    SSS             CCC             AAA       AAA   LLL
PPP         PPP    AAA       AAA    SSS             CCC             AAA       AAA   LLL
PPP         PPP    AAA       AAA    SSS             CCC             AAA       AAA   LLL
PPP         PPP    AAA       AAA    SSS             CCC             AAA       AAA   LLL
PPPPPPPPPPPP       AAA       AAA      SSSSSSSS      CCC             AAA       AAA   LLL
PPPPPPPPPPPP       AAA       AAA      SSSSSSSS      CCC             AAA       AAA   LLL
PPPPPPPPPPPP       AAA       AAA      SSSSSSSS      CCC             AAA       AAA   LLL
PPP                AAAAAAAAAAAAAAA          SSS     CCC             AAAAAAAAAAAAAAA   LLL
PPP                AAAAAAAAAAAAAAA          SSS     CCC             AAAAAAAAAAAAAAA   LLL
PPP                AAAAAAAAAAAAAAA          SSS     CCC             AAAAAAAAAAAAAAA   LLL
PPP                AAA       AAA            SSS     CCC             AAA       AAA     LLL
PPP                AAA       AAA            SSS     CCC             AAA       AAA     LLL
PPP                AAA       AAA            SSS     CCC             AAA       AAA     LLL
PPP                AAA       AAA    SSSSSSSSSSS       CCCCCCCCCCC   AAA       AAA     LLLLLLLLLLLLLLL
PPP                AAA       AAA    SSSSSSSSSSS       CCCCCCCCCCC   AAA       AAA     LLLLLLLLLLLLLLL
PPP                AAA       AAA    SSSSSSSSSSS       CCCCCCCCCCC   AAA       AAA     LLLLLLLLLLLLLLL
```

```
PPPPPPPP      AAAAAA      SSSSSSSS  RRRRRRRR    TTTTTTTTTT    333333
PPPPPPPP      AAAAAA      SSSSSSSS  RRRRRRRR    TTTTTTTTTT    333333
PP       PP  AA      AA  SS        RR      RR      TT        33      33
PP       PP  AA      AA  SS        RR      RR      TT        33      33
PP       PP  AA      AA  SS        RR      RR      TT                33
PP       PP  AA      AA  SS        RR      RR      TT                33
PPPPPPPP     AA      AA    SSSSSS  RRRRRRRR        TT              33
PPPPPPPP     AA      AA    SSSSSS  RRRRRRRR        TT              33
PP           AAAAAAAAAA          SS  RR  RR        TT                33
PP           AAAAAAAAAA          SS  RR  RR        TT                33
PP           AA      AA          SS  RR    RR      TT        33      33      ....
PP           AA      AA          SS  RR    RR      TT        33      33      ....
PP           AA      AA  SSSSSSSS    RR      RR    TT          333333        ....
PP           AA      AA  SSSSSSSS    RR      RR    TT          333333        ....

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 ;
0000     2 ;*****************************************************************************
0000     3 ;*                                                                           *
0000     4 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     5 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     6 ;*  ALL RIGHTS RESERVED.                                                     *
0000     7 ;*                                                                           *
0000     8 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000     9 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    10 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    11 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    12 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    13 ;*  TRANSFERRED.                                                             *
0000    14 ;*                                                                           *
0000    15 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    16 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    17 ;*  CORPORATION.                                                             *
0000    18 ;*                                                                           *
0000    19 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    20 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    21 ;*                                                                           *
0000    22 ;*                                                                           *
0000    23 ;*****************************************************************************
0000    24 ;*                                                                           *
0000    25 ;*                              PAS$RT_CHK                                   *
0000    26 ;*                RUNTIME SUPPORT MODULE FOR PASCAL -- SECTION 3           *
0000    27 ;*                                                                           *
0000    28 ;*                          VERSION V1.0-1 - OCTOBER 1979                    *
0000    29 ;*                                                                           *
0000    30 ;*    This module defines the following routines:                            *
0000    31 ;*                                                                           *
0000    32 ;*       pas$okset:   routine to do runtime set size checking                *
0000    33 ;*       pas$asserr:  routine to report a subrange assignment error          *
0000    34 ;*       pas$seterr:  routine to report a set assignment error               *
0000    35 ;*       pas$diverr:  routine to report a divide-by-zero error               *
0000    36 ;*       pas$dynerr:  routine to report a dynamic array assignment error     *
0000    37 ;*       pas$caserr:  routine to report case range error                     *
0000    38 ;*                                                                           *
0000    39 ;*    Written by: Jeff Scofield 10-Dec-78                                    *
0000    40 ;*                Hellmut Golde 15-Feb-79                                    *
0000    41 ;*                Jan Sanislo   22-Feb-79                                    *
0000    42 ;*                                                                           *
0000    43 ;*    edit history: 01-002 26feb80 paul hohensee. inhibit message print
0000    44 ;*                  when signalling error.
0000    45 ;*
0000    46 ;*                  01-003 12may80 susan azibert.  Change the output of
0000    47 ;*                  run-time error messages so that they are signalled
0000    48 ;*                  and printed by calling lib$stop.
0000    49 ;
0000    50 ;                 01-004 Paul Hohensee 11-Aug-81. Change to general addressing of exte
0000    51 ;*                                                                           *
0000    52 ;*****************************************************************************
0000    53          .title  pas$rt_chk
0000    54          .ident  'V04-000'
00000000    55          .psect _pas$code,pic,shr,exe,nowrt
0000    56
0000    57          .extrn  pas$_subasgbou  ; message 8120
```

```
                                0000     58          .extrn  pas$_setasgbou  ; message 8130
                                0000     59          .extrn  pas$_invasginc  ; message 8150
                                0000     60          .extrn  pas$_casselbou  ; message 8140
                                0000     61
                                0000     62          $stsdef                 ; status code macros
                                0000     63 ;
                                0000     64 ;   ROUTINE TO IMPLEMENT RUNTIME SET SIZE CHECKING
                                0000     65 ;
                          0004  0000     66          .entry  pas$okset,^m<r2>
                     04   AC     DD  0002     67          pushl   4(ap)
                     0C   AC     DD  0005     68          pushl   12(ap)
          00000000'GF        02   FB  0008     69          calls   #2,G^pas$card   ; check low end of set
                             50   D5  000F     70          tstl    r0
                             23   12  0011     71          bneq    10$             ; out of bounds if not zero
                     04   AC     DD  0013     72          pushl   4(ap)
                     10   AC     DD  0016     73          pushl   16(ap)
          00000000'GF        02   FB  0019     74          calls   #2,G^pas$card   ; check total range
                        52   50   D0  0020     75          movl    r0,r2           ; save result
                     04   AC     DD  0023     76          pushl   4(ap)
                     08   AC     DD  0026     77          pushl   8(ap)
          00000000'GF        02   FB  0029     78          calls   #2,G^pas$card   ; check total range except high end
                        52   50   C2  0030     79          subl2   r0,r2           ; find cardinality of upper end
                             01   12  0033     80          bneq    10$             ; out of bounds if not zero
                                  04  0035     81          ret
          0000005B'EF        00   FB  0036     82 10$:     calls   #0,pas$seterr   ; call set error routine
                                  04  003D     83          ret
                                      003E     84 ;
                                      003E     85 ;   ROUTINE TO IMPLEMENT THE PROCEDURE PAS$ASSERR
                                      003E     86 ;
                                0000  003E     87          .entry  pas$asserr,^m<>
                7E   10 AD      07    C3  0040     88          subl3   #7,16(fp),-(sp)         ; third FAO argument (PC of call)
                                00    DD  0045     89          pushl   #0                      ; second FAO argument (null)
                                00    DD  0047     90          pushl   #0                      ; first FAO argument (null)
                                03    DD  0049     91          pushl   #3                      ; number of FAO arguments preceding
7E   00000000'8F                04    C1  004B     92          addl3   #4,#pas$_subasgbou,-(sp)        ; push error message #8120
     00000000'GF                05    FB  0053     93          calls   #5,G^lib$stop           ; signal error and stop execution
                                      04  005A     94          ret
                                      005B     95 ;
                                      005B     96 ;   ROUTINE TO IMPLEMENT THE PROCEDURE PAS$SETERR
                                      005B     97 ;
                                0000  005B     98          .entry  pas$seterr,^m<>
                7E   10 AD      07    C3  005D     99          subl3   #7,16(fp),-(sp)         ; third FAO argument (PC of call)
                                00    DD  0062    100          pushl   #0                      ; second FAO argument (null)
                                00    DD  0064    101          pushl   #0                      ; first FAO argument (null)
                                03    DD  0066    102          pushl   #3                      ; count of FAO arguments preceding
7E   00000000'8F                04    C1  0068    103          addl3   #4,#pas$_setasgbou,-(sp)        ; push error message #8130
     00000000'GF                05    FB  0070    104          calls   #5,G^lib$stop           ; signal error and stop execution
                                      04  0077    105          ret
                                      0078    106 ;
                                      0078    107 ;   ROUTINE TO IMPLEMENT THE PROCEDURE PAS$DIVERR
                                      0078    108 ;
                                      0078    109          $ssdef
                                      0078    110 ;
                                0000  0078    111          .entry  pas$diverr,^m<>
                00000484 8F           DD  007A    112          pushl   #ss$_intdiv
     00000000'GF                01    FB  0080    113          calls   #1,G^lib$stop
                                      04  0087    114          ret
```

```
                               0088   ;
                               0088   ;  ROUTINE TO IMPLEMENT THE PROCEDURE PAS$DYNERR
                               0088   ;
                         0000  0088        .entry  pas$dyneer,^m<>
      7E    10 AD    07    C3  008A        subl3   #7,16(fp),-(sp)                ; third FAO argument (PC of call)
                     00    DD  008F        pushl   #0                             ; second FAO argument (null)
                     00    DD  0091        pushl   #0                             ; first FAO argument (null)
                     03    DD  0093        pushl   #3                             ; number of FAO arguments preceding
      7E  00000000'8F    04    C1  0095    addl3   #4,#pas$_invasginc,-(sp)       ; push error message #8150
          00000000'GF    05    FB  009D    calls   #5,G^lib$stop                  ; signal error and stop execution
                           04  00A4        ret
                               00A5   ;
                               00A5   ;  ROUTINE TO IMPLEMENT THE PROCEDURE PAS$CASERR
                               00A5   ;
                         0000  00A5        .entry  pas$caserr,^m<>
      7E    10 AD    07    C3  00A7        subl3   #7,16(fp),-(sp)                ; third FAO argument (PC of call)
                     00    DD  00AC        pushl   #0                             ; second FAO argument (null)
                     00    DD  00AE        pushl   #0                             ; first FAO argument (null)
                     03    DD  00B0        pushl   #3                             ; count of FAO arguments preceding
      7E  00000000'8F    04    C1  00B2    addl3   #4,#pas$_casselbou,-(sp)       ; push error message #8140
          00000000'GF    05    FB  00BA    calls   #5,G^lib$stop                  ; signal error and stop execution
                           04  00C1        ret
                               00C2        .END
```

```
LIB$STOP                        ********    X   01
PAS$ASSERR                      0000003E RG     01
PAS$CARD                        ********    X   01
PAS$CASERR                      000000A5 RG     01
PAS$DIVERR                      00000078 RG     01
PAS$DYNERR                      00000088 RG     01
PAS$OKSET                       00000000 RG     01
PAS$SETERR                      0000005B RG     01
PAS$_CASSELBOU                  ********    X   01
PAS$_INVASGINC                  ********    X   01
PAS$_SETASGBOU                  ********    X   01
PAS$_SUBASGBOU                  ********    X   01
SS$_INTDIV                    = 00000484
```

Psect synopsis

```
PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD  NOWRT NOVEC BYTE
_PAS$CODE                       000000C2  (  194.)  01 (  1.)    PIC   USR   CON   REL   LCL   SHR   EXE   RD  NOWRT NOVEC BYTE
$ABS$                           00000000  (    0.)  02 (  2.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
```

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	33	00:00:00.08	00:00:00.97
Command processing	110	00:00:00.43	00:00:03.46
Pass 1	187	00:00:04.19	00:00:08.72
Symbol table sort	0	00:00:00.66	00:00:01.70
Pass 2	42	00:00:00.78	00:00:02.71
Symbol table output	3	00:00:00.02	00:00:00.02
Psect synopsis output	3	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	381	00:00:06.21	00:00:17.62

The working set limit was 900 pages.
22350 bytes (44 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 451 non-local and 1 local symbols.
137 source lines were read in Pass 1, producing 28 object records in Pass 2.
9 pages of virtual memory were used to define 8 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	5

505 GETS were required to define 5 macros.

There were no errors, warnings or information messages.

```
MACRO/DISABLE=TRACE/LIS=LIS$:PASRT3/OBJ=OBJ$:PASRT3 MSRC$:PASRT3/UPDATE=(ENH$:PASRT3)
```

PASRT1
LIS

PASMACROS
REQ

PASFCB
SDL

PASPROLOG
REQ

PASOPEDEF
REQ

PASABSL
LIS

PASCARD2
LIS

PASRTL

PASBUGCOD
REQ

PASCLOSE2
LIS

PASRT2
LIS

PASRT3
LIS

PASRTL
MAP

PASIO4
LIS

PASEXTERN
REQ

PASKDB
REQ

PASPFD
REQ

PASCONVER
LIS

PASLIB
REQ

PASPFV
REQ

PASBIN
LIS

PASCLOCK2
LIS

PASRT4
LIS