```
MMM          MMM  TTTTTTTTTTTTTTT   HHH          HHH  RRRRRRRRRRR     TTTTTTTTTTTTTTT   LLL
MMM          MMM  TTTTTTTTTTTTTTT   HHH          HHH  RRRRRRRRRRRR    TTTTTTTTTTTTTTT   LLL
MMM          MMM  TTTTTTTTTTTTTTT   HHH          HHH  RRRRRRRRRRRR    TTTTTTTTTTTTTTT   LLL
MMMMMM    MMMMMM        TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMMMMM    MMMMMM        TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMMMMM    MMMMMM        TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM   MMM   MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM   MMM   MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM   MMM   MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM         MMM         TTT         HHHHHHHHHHHHHHHH  RRRRRRRRRRR           TTT         LLL
MMM         MMM         TTT         HHHHHHHHHHHHHHHH  RRRRRRRRRRRR          TTT         LLL
MMM         MMM         TTT         HHHHHHHHHHHHHHHH  RRRRRRRRRRRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR   RRR             TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR   RRR             TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR   RRR             TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR      RRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR      RRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR      RRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLLLLLLLLLLLLLLL
MMM         MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLLLLLLLLLLLLLLL
MMM         MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLLLLLLLLLLLLLLL
```

```
MM      MM  TTTTTTTTTT  HH      HH    CCCCCCCC    SSSSSSSS    QQQQQQ    RRRRRRRR    TTTTTTTTTT
MM      MM  TTTTTTTTTT  HH      HH    CCCCCCCC    SSSSSSSS    QQQQQQ    RRRRRRRR    TTTTTTTTTT
MMMM  MMMM      TT      HH      HH  CC          SS          QQ    QQ  RR      RR      TT
MMMM  MMMM      TT      HH      HH  CC          SS          QQ    QQ  RR      RR      TT
MM  MM  MM      TT      HH      HH  CC          SS          QQ    QQ  RR      RR      TT
MM  MM  MM      TT      HH      HH  CC          SS          QQ    QQ  RR      RR      TT
MM      MM      TT      HHHHHHHHHH  CC            SSSSSS    QQ    QQ  RRRRRRRR        TT
MM      MM      TT      HHHHHHHHHH  CC            SSSSSS    QQ    QQ  RRRRRRRR        TT
MM      MM      TT      HH      HH  CC                  SS  QQ  QQ  QQ  RR  RR          TT
MM      MM      TT      HH      HH  CC                  SS  QQ  QQ  QQ  RR  RR          TT
MM      MM      TT      HH      HH  CC                  SS  QQ    QQ    RR    RR        TT
MM      MM      TT      HH      HH  CC                  SS  QQ    QQ    RR    RR        TT
MM      MM      TT      HH      HH    CCCCCCCC  SSSSSSSS      QQQQ  QQ  RR      RR      TT
MM      MM      TT      HH      HH    CCCCCCCC  SSSSSSSS      QQQQ  QQ  RR      RR      TT
```

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

Table of contents

(2) 48 HISTORY ; Detailed Current Edit History
(3) 60 DECLARATIONS
(4) 87 MTH$CSQRT - compute COMPLEX square root

```
0000     1          .TITLE  MTH$CSQRT
0000     2          .IDENT  /1-005/         ; File: MTHCSQRT.MAR  Edit: SBL1005
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 ; FACILITY: MATH LIBRARY
0000    30 ;++
0000    31 ; ABSTRACT:
0000    32 ;       This module contains routine MTH$CSQRT - compute COMPLEX square root.
0000    33 ;
0000    34 ;--
0000    35 ;
0000    36 ; VERSION: 0
0000    37 ;
0000    38 ; HISTORY:
0000    39 ;
0000    40 ; AUTHOR:
0000    41 ;       Jonathan M. Taylor, 20-JUL-77: Version 0
0000    42 ;
0000    43 ; MODIFIED BY:
0000    44 ;
0000    45 ;
0000    46 ;
```

```
0000     48          .SBTTL  HISTORY         ; Detailed Current Edit History
0000     49
0000     50
0000     51 ; Edit History for Version 0 of MTH$CSQRT
0000     52 ;
0000     53 ; 0-3    - Fix comments.  TNH 16-June-78
0000     54 ; 1-001  - Update version number and copyright notice.  JBS 16-NOV-78
0000     55 ; 1-002  - Add "_" to the PSECT directives.  JBS 21-DEC-78
0000     56 ; 1-003  - Fix zerodivide bug on (0,0).  SPR 22832  SBL 2-Mar-79
0000     57 ; 1-004  - Use MTH$SQRT_R3.  SBL 27-Sept-1979
0000     58 ; 1-005  - Use general mode addressing.  SBL 30-Nov-1981
```

```
                     60          .SBTTL  DECLARATIONS
                     61
                     62 ;
                     63 ; INCLUDE FILES:
                     64 ;
                     65
                     66 ;
                     67 ; EXTERNAL SYMBOLS:
                     68          .GLOBL  MTH$CABS
                     69          .GLOBL  MTH$SQRT_R3
                     70
                     71 ;
                     72 ; MACROS:
                     73 ;        NONE
                     74
                     75 ;
                     76 ; PSECT DECLARATIONS:
00000000             77          .PSECT  _MTH$CODE       PIC, SHR, LONG, EXE, NOWRT
                     78
                     79 ;
                     80 ; EQUATED SYMBOLS:
00000004             81          argadr  =       4       ; offset from AP of arg address
                     82
                     83 ;
                     84 ; OWN STORAGE:
                     85 ;        NONE
```

```
                        .SBTTL  MTH$CSQRT - compute COMPLEX square root

                ;++
                ; FUNCTIONAL DESCRIPTION:
                ;
                ;       The square root of a complex number (r, i) is computed
                ;       as follows:
                ;
                ;       ROOT = SQRT((ABS(r) + CABS((r, i))) / 2)
                ;       Q = i / (2*ROOT)
                ;
                ;
                ;       r       i       CSQRT((r, i))
                ;       -       -       -------------
                ;
                ;       >=0     any     (ROOT, Q)
                ;       <0      >=0     (Q, ROOT)
                ;       <0      <0      (-Q, -ROOT)
                ;
                ; CALLING SEQUENCE:
                ;
                ;       Square_root.wfc.v       = MTH$CSQRT (arg.rfc.r)
                ;
                ; INPUT PARAMETERS:
                ;
                ;       The one input parameter is the address of a COMPLEX number (r, i)
                ;       where r and i are both single-precision floating point values.
                ;
                ; IMPLICIT INPUTS:
                ;       NONE
                ;
                ; OUTPUT PARAMFTERS:
                ;       NONE
                ;
                ; IMPLICIT OUTPUTS:
                ;       NONE
                ;
                ; COMPLETION CODES:
                ;       NONE
                ;
                ; SIDE EFFECTS:
                ;
                ;       Signals:        Reserved Operand if r or i is bad (-0.0)
                ;
                ;--

                .ENTRY  MTH$CSQRT,      ^M<R2, R3>
                MTH$FLAG_JACKET                 ; flag as math routine

6D  00000000'GF  9E             MOVAB   G^MTH$$JACKET_HND, (FP)
                                                ; set handler address to jacket
                                                ; handler

                        MOVF    @argadr(AP), R2         ; R2 = r
    52  04 BC  50          MOVF    @argadr(AP), R2         ; R2 = r
52  8000 8F  AA             BICW    #^X8000, R2             ; R2 = ABS(r)
```

```
                00000000'GF   6C   FA  0012   139          CALLG   (AP), G^MTH$CABS        ; R0 = CABS((r, i))
                        50   52   40  0019   140          ADDF    R2, R0                  ; R0 = ABS(r) + CABS((r, i))
                        50   00   44  001C   141          MULF    #0.5, R0                ; R0 = (ABS(r) + CABS((r, i))) / 2
                00000000'GF   16   001F   142          JSB     G^MTH$SQRT_R3           ; R0 = ROOT = SQRT(above)
                     52   04   AC   D0  0025   143          MOVL    argadr(AP), R2          ; R2 -> (r, i)
                          50   53  0029   144          TSTF    R0                      ; is ROOT zero?
                          04   12  002B   145          BNEQ    1$                      ; no, go ahead
                          51   D4  002D   146          CLRL    R1                      ; make zero quotient
                          08   11  002F   147          BRB     2$                      ; skip divide
                51   04   A2   50  47  0031   148 1$:      DIVF3   R0, 4(R2), R1           ; R1 = i / ROOT
                        51   00  44  0036   149          MULF    #0.5, R1                ; R1 = Q = i / (2 * ROOT)
                          82   53  0039   150 2$:       TSTF    (R2)+                   ; if r positive,
                          14   18  003B   151          BGEQ    RETRN                   ; then return (ROOT, Q)
                        53   50  D0  003D   152          MOVL    R0, R3                  ; else switch ROOT and Q
                          62   53  0040   153          TSTF    (R2)                    ; if i positive
                          07   18  0042   154          BGEQ    RETRN1                  ; then return (Q, ROOT)
                        50   51  52  0044   155          MNEGF   R1, R0                  ; else negate ROOT and Q
                        51   53  52  0047   156          MNEGF   R3, R1                  ; and return (-Q, -ROOT)
                             04  004A   157          RET
                                 004B   158
                                 004B   159 RETRN1:
                        50   51  D0  004B   160          MOVL    R1, R0                  ; continue to swap ROOT and Q
                        51   53  D0  004E   161          MOVL    R3, R1                  ; and return (Q, ROOT)
                                 0051   162 RETRN:
                             04  0051   163          RET
                                 0052   164
                                 0052   165
                                 0052   166          .END
```

```
ARGADR          = 00000004
MTH$$JACKET_HND  ********    X   01
MTH$CABS         ********  G     00
MTH$CSQRT        00000000 RG    01
MTH$SQRT_R3      ********  G     00
RETRN            00000051 R     01
RETRN1           0000004B R     01
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
_MTH$CODE	00000052 (82.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	39	00:00:00.15	00:00:00.78
Command processing	127	00:00:00.70	00:00:06.16
Pass 1	83	00:00:00.69	00:00:02.25
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	45	00:00:00.49	00:00:02.20
Symbol table output	2	00:00:00.02	00:00:00.02
Psect synopsis output	2	00:00:00.02	00:00:00.07
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	300	00:00:02.09	00:00:11.49

The working set limit was 750 pages.
2731 bytes (6 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 7 non-local and 2 local symbols.
226 source lines were read in Pass 1, producing 11 object records in Pass 2.
1 page of virtual memory was used to define 1 macro.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[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$:MTHCSQRT/OBJ=OBJ$:MTHCSQRT MSRC$:MTHJACKET/UPDATE=(ENH$:MTHJACKET)+MSRC
```

MTHCVTDG
LIS

MTHDACOS
LIS

MTHCGABS
LIS

MTHCDSINC
LIS

MTHCLOG
LIS

MTHDASIN
LIS

MTHCGLOG
LIS

MTHCONVER
LIS

MTHDATAN
LIS

MTHCSQRT
LIS

MTHCEXP
LIS

MTHCGSQRT
LIS

MTHCGEXP
LIS

MTHCSINCO
LIS

MTHCONJG
LIS

MTHCDSQRT
LIS

MTHCGSINC
LIS

MTHCOSH
LIS