

[illegible]

3

Sy

MT

MT
MT

MT

MT

MT
MTMT
MTMT
MTMT
MT

MT
MT

MT
MT

MT

MT

MT
MT

MI
MT

MT
MT

MT
MT

MT
MT

MT
MT

MT

111

227

MT

MT
MT

MT

MT

M1
M1MT
MT

M1

M1

10

1

10

100

10

1

```

MM      MM      TTTTTTTTTT  HH      HH      SSSSSSSS  QQQQQQ  RRRRRRRR  TTTTTTTTTT  RRRRRRRR  222222
MM      MM      TTTTTTTTTT  HH      HH      SSSSSSSS  QQQQQQ  RRRRRRRR  TTTTTTTTTT  RRRRRRRR  222222
MMM     MMM     TT          HH      HH      SS          QQ      QQ      RR      RR      22      22
MMM     MMM     TT          HH      HH      SS          QQ      QQ      RR      RR      22      22
MM      MM      TT          HH      HH      SS          QQ      QQ      RR      RR      22      22
MM      MM      TT          HH      HH      SS          QQ      QQ      RR      RR      22      22
MM      MM      TT          HHHHHHHHHHH  SSSSSS  QQ      QQ      RRRRRRRR  TT      RRRRRRRR  22
MM      MM      TT          HHHHHHHHHHH  SSSSSS  QQ      QQ      RRRRRRRR  TT      RRRRRRRR  22
MM      MM      TT          HH      HH      SS          QQ      QQ      RR      RR      22      22
MM      MM      TT          HH      HH      SS          QQ      QQ      RR      RR      22      22
MM      MM      TT          HH      HH      SS          QQ      QQ      RR      RR      22      22
MM      MM      TT          HH      HH      SS          QQ      QQ      RR      RR      22      22
MM      MM      TT          HH      HH      SSSSSSSS  QQ      QQ      RR      RR      2222222222
MM      MM      TT          HH      HH      SSSSSSSS  QQ      QQ      RR      RR      2222222222

```

```

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
LLLLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLLLL  IIIIII  SSSSSSSS

```


(2) 52
(3) 79
(4) 117
(5) 192

HISTORY : Detailed Current Edit History
DECLARATIONS : Declarative Part of Module
MTH\$SQRT_R2 - Standard Single Precision Floating SQRT
MTH\$SQRT_R2 - JSB SQRT routine

```
0000 1 .TITLE MTH$SQRT2 ; Floating Point Square Root routine
0000 2 ; (SQRT)
0000 3 .IDENT /1-015/ ; File: MTH$SQRT2.MAR EDIT: RNH1015
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
0000 33 MTH$SQRT_R2 is a special routine which is the same as MTH$SQRT except
0000 34 a faster non-standard JSB call is used with the argument in R0 and no
0000 35 registers are saved.
0000 36
0000 37 --
0000 38
0000 39 VERSION: 01
0000 40
0000 41 HISTORY:
0000 42 AUTHOR:
0000 43 Peter Yuo, 15-Oct-76: Version 01
0000 44
0000 45 MODIFIED BY:
0000 46
0000 47 01-1 Peter Yuo, 22-May-77
0000 48 01-2 Peter Yuo, 31-May-77
0000 49
0000 50 ;
```



```
0000 52      .SBTTL HISTORY ; Detailed Current Edit History
0000 53
0000 54
0000 55 : ALGORITHMIC DIFFERENCES FROM FP-11/C ROUTINE: none
0000 56
0000 57 : Edit History for Version 01 of MTH$SQRT
0000 58
0000 59 : 01-1 Code saving after code review
0000 60 : 01-2 ROTL shift in garbage into highest bit. Use ASHL instead.
0000 61 : ADDL instruction after ADJUST has been changed into ADDW to prevent
0000 62 : overflow if R1<31:16> = FFFF and R0<31:16> = FFFF
0000 63 : 01-3 Finish error handling 10-June-1977
0000 64 : 01-5 MTH$ERROR changed to MTH$SIGNAL.
0000 65 : MTH$... changed to MTH$....
0000 66 : Changed error handling mechanism. Put error result in R0 before
0000 67 : calling MTH$SIGNAL in order to allow user modify error result.
0000 68 : 01-6 Return -0.0 on negative arg. TNH 20-Dec-77
0000 69 : 01-7 Edit in Rich Lary's code bums. JSB routine is now R2. JMT 19-Jan-78
0000 70 : 01-9 Move .ENTRY symbol to module header. TNH 14-Aug-78
0000 71 : 1-010 - Put version number in standard format: three digit edit
0000 72 : numbers. Also, update the copyright notice. JBS 16-NOV-78
0000 73 : 1-011 - Change MTH_SQUROONEG to MTH$K_SQUROONEG. JBS 07-DEC-78
0000 74 : 1-012 - Add " " to the PSECT directive. JBS 22-DEC-78
0000 75 : 1-013 - Declare externals. SBL 17-May-1979
0000 76 : 1-014 - Move MTH$SQRT_R2 to separate module. JAW 26-Sep-1979.
0000 77 : 1-015 - Change external references to G^ RNH 06-Oct-81
```

```
0000 79      .SBTTL  DECLARATIONS      ; Declarative Part of Module
0000 80
0000 81 :
0000 82 : INCLUDE FILES:
0000 83 :
0000 84 :
0000 85 :
0000 86 : EXTERNAL SYMBOLS:
0000 87 :
0000 88      .DSABL  GBL
0000 89      .EXTRN  MTH$K_SQUROONEG
0000 90      .EXTRN  MTH$$SIGNAL
0000 91
0000 92 :
0000 93 : EQUATED SYMBOLS:
0000 94 :
0000 95 : MACROS:      none
0000 96 :
0000 97 : PSECT DECLARATIONS:
0000 98
0000 99      .PSECT  _MTH$CODE      PIC,SHR,LONG,EXE,NOWRT
0000 100      ; program section for math routines
0000 101 :
0000 102 : OWN STORAGE:  none
0000 103 :
0000 104 : CONSTANTS:
0000 105 :
0000 106 :
0000 107 : Constants A and B chosen for k = odd
0000 108 :
13CD5FD4 0000 109      LF_ODD_A_E63      =      ^X13CD5FD4
3C4A2018 0000 110      LF_ODD_B_EM63      =      ^X3C4A2018
0000 111 :
0000 112 : Constants A and B chosen for k = even
0000 113 :
F61A4015 0000 114      LF_EVEN_A      =      ^XF61A4015
4B231FD7 0000 115      LF_EVEN_B_EM64      =      ^X4B231FD7
```



```
0000 117 .SBTTL MTH$SQRT2 - Standard Single Precision Floating SQRT
0000 118
0000 119
0000 120 :++
0000 121 : FUNCTIONAL DESCRIPTION:
0000 122
0000 123 : SQRT - single precision floating point function
0000 124
0000 125 : SQRT(X) is computed using the following approximation technique:
0000 126
0000 127 : If X <= 0 , error. Let X = |X|.
0000 128
0000 129 : Let X = 2**K * F where F is the fractional part.
0000 130
0000 131 : If K = even, X = 2**(2P) * F,
0000 132 : SQRT(X) = 2**P * SQRT(F), 1/2 <= F < 1
0000 133
0000 134 : If K = odd, X = 2**(2P+1) * F = 2**(2P+2) * (F/2),
0000 135 : SQRT(X) = 2**(P+1) * SQRT(F/2), 1/4 <= F/2 < 1/2.
0000 136
0000 137 : Let F' = A*F + B,
0000 138 : A = 0.453730314(octal),
0000 139 : B = 0.327226214(octal), for K = even.
0000 140 : = A*(F/2) + B,
0000 141 : A = 0.650117146(octal),
0000 142 : B = 0.230170444(octal), for K = odd.
0000 143
0000 144 : and
0000 145 : K' = P, for K = even
0000 146 : = P + 1 for K = odd.
0000 147
0000 148 : Let Y0 = 2**K' * F' as a straight line approximation within the
0000 149 : given interval using coefficients A and B which minimize the
0000 150 : absolute error at the midpoint and endpoint.
0000 151
0000 152 : Starting with Y0, two Newton-Raphson iterations are performed.
0000 153
0000 154 : Y[n+1] = (1/2) * ( Y[n] + X/Y[n] )
0000 155
0000 156 : The relative error is < 10**-8.
0000 157 : CALLING SEQUENCE:
0000 158
0000 159 : sqrt.wf.v = MTH$SQRT2(x.rf.r)
0000 160
0000 161 : INPUT PARAMETERS:
0000 162
0000 163 : LONG = 4 ; define longword multiplier
0000 164 : x = 1 * LONG ; Contents of x is the argument
0000 165
0000 166 : IMPLICIT INPUTS: none
0000 167
0000 168 : OUTPUT PARAMETERS:
0000 169
0000 170 : VALUE: floating square root of the argument
0000 171
0000 172 : IMPLICIT OUTPUTS: none
0000 173 :
```

00000004
00000004


```

0000 174 : COMPLETION CODES:      none
0000 175 :
0000 176 : SIDE EFFECTS:
0000 177 :
0000 178 : Signals: MTH$_SQUROONEG if X < 0.0 with reserved operand in R0 (copied to
0000 179 : the signal mechanism vector CHF$L_MCH_R0/R1 by LIB$SIGNAL).
0000 180 : Associated message is: "SQUARE ROOT OF NEGATIVE VALUE". Result is reserved
0000 181 : operand -0.0 unless a user supplied (or any) error handler changes CHF$L_MCH_R0/R1
0000 182 :
0000 183 : NOTE: This procedure disables floating point underflow, enables integer
0000 184 : overflow, causes no floating overflow or other arithmetic traps, and
0000 185 : preserves enables across the call.
0000 186 :
0000 187 :---
0000 188 :
0000 189 :
0000 190 :

```



```
0000 192 .SBTTL MTH$SQRT_R2 - JSB SQRT routine
0000 193
0000 194 ; JSB SQRT - used by the standard, and directly.
0000 195
0000 196 CALLING SEQUENCE:
0000 197 ; save anything in R0:R2
0000 198 ; MOVF R0 ; input in R0
0000 199 ; JSB MTH$SQRT_R2
0000 200 ; return with result in R0
0000 201
0000 202 ; Note: This routine is written to avoid any integer overflows, floating overflows,
0000 203 ; floating underflows or divide by 0 conditions, whether enabled or not.
0000 204
0000 205 ; REGISTERS USED:
0000 206 ; R0 - Floating argument then result
0000 207 ; R1 - X saved for use during iteration
0000 208 ; R2 - scratch
0000 209
0000 210 MTH$SQRT_R2:: ; JSB routine for SQRT
51 50 50 0000 211 MOVF R0, R1 ; test sign of X and save it in R1.
55 15 0003 212 BLEQ ZERO_NEG ; branch to ZERO_NEG if X <= 0
0005 213
0005 214 ; X > 0
0005 215
0005 216 POS:
0005 217 CLRL -(SP) ; make room for 2nd half of
0007 218 ; double length operand
52 50 3C 0007 219 MOVZWL R0, R2 ; isolate low 16 bits (sign, exp, > fract) in R
52 94 000A 220 CLRB R2 ; R2 now has sign and left 7 exp bits
50 52 AA 000C 221 BICW R2, R0 ; clear sign and left 7 exp bits
50 95 000F 222 TSTB R0 ; check low bit of exp
50 10 18 0011 223 BGEQ EVEN ; and branch if 1
50 13CD5FD4 8F 44 0013 224 MULF #LF_ODD_A_E63, R0 ; add 64 (half of bias) to (exponent-2)
50 3C4A2018 8F 40 001A 225 ADDF #LF_ODD_B_EM63, R0 ; and start approximation calc
13 11 0021 226 BRB ADJUST ; R0 = (first approx) * 2**(-64)
0023 227 ; go adjust
0023 228
0023 229 EVEN:
50 2000 8F A0 0023 230 ADDW #X2000, R0 ; exp is 0 - make it 64 (2**(-64)) for legalit
50 F61A4015 8F 44 0028 231 MULF #LF_EVEN_A, R0
50 4B231FD7 8F 40 002F 232 ADDF #LF_EVEN_B_EM64, R0 ; R0 = (first approx) * 2**(-64)
0036 233 ADJUST:
52 52 1F 9C 0036 234 ROTL #31, R2, R2 ; divide R2 (exp+bias) by 2,
003A 235 ; giving (exp/2+64)
50 52 A0 003A 236 ADDW R2, R0 ; insert exp/2 in first approx and
003D 237 ; re-bias it.
003D 238
003D 239 ; first iteration - single precision is sufficient
003D 240
52 51 50 47 003D 241 DIVF3 R0, R1, R2 ; R2 = X/Y0
50 50 52 40 0041 242 ADDF R2, R0 ; R0 = Y0 + X/Y0
50 0080 8F A2 0044 243 SUBW #X80, R0 ; R0 = Y1 = (1/2)(Y0 + X/Y0)
0049 244 ; no overflow possible
0049 245
0049 246 ; second iteration, do in double precision to get truncated( rather than
0049 247 ; rounded) result.
0049 248 ;
```

```
0049 249 ::: CLRL R2 ; lower part (X) = 0
0049 250 ::: DIVD R0, R1 ; divide Y1 into X with low-order
0049 251 ::: ; 32 bits of Y1 garbage. This doesn't
0049 252 ; effect accuracy, since Y1 innacurate
0049 253 ; anyway.
51 51 DD 0049 254 PUSHL R1 ; convert x and place on stack
51 51 D4 004B 255 CLRL R1 ; clear low part of Y1
51 8E 50 67 004D 256 DIVD3 R0, (SP)+, R1 ; divide Y1 into X
50 50 51 40 0051 257 ADDF R1, R0 ; R0 = Y1 + higher part(X/Y1)
50 0080 8F A2 0054 258 SUBW #^X80, R0 ; R0 = SQ(X) = (T/2) (Y1 + X/Y1)
05 0059 259 SQRTX: RSB ; return, R0 = result
005A 260
005A 261 ; X =< 0
005A 262 ;
005A 263 ZERO_NEG:
005A 264 BEQL SQRTX ; return with R0 = result = 0
005C 265 PUSHL (SP) ; return PC from JSB routine
7E 00 8F 9A 005E 266 MOVZBL #MTH$K_SQURONEG, -(SP) ; condition value
50 01 0F 78 0062 267 ASHL #15, #T, R0 ; R0 = result = reserved operand -0.0
0066 268 ; R0 goes to signal mechanism vector
0066 269 ; (CHF$MCH_R0/R1) so error handler
0066 270 ; can modify the result.
00000000'GF 02 FB 0066 271 CALLS #2, G^MTH$$SIGNAL ; signal error and use real user's PC
006D 272 ; independent of CALL vs JSB
05 006D 273 RSB ; return - R0 restored from CHF$MCH_R0/R1
006E 274
006E 275 .END
```


MTH\$SQRT2
Symbol table

F 1
; Floating Point Square Root routine

16-SEP-1984 01:51:35
6-SEP-1984 11:27:15

VAX/VMS Macro V04-00
[MTHRTL.SRC]MTH\$SQRT2.MAR;1

Page 8
(5)

ADJUST	00000036	R	01
EVEN	00000023	R	01
LF_EVEN_A	= F61A4015		
LF_EVEN_B	EM64 = 4B231FD7		
LF_ODD_A	E63 = 13CD5FD4		
LF_ODD_B	EM63 = 3C4A2018		
LONG	= 00000004		
MTH\$SIGNAL	*****	X	00
MTH\$K_SQUROONEG	*****	X	00
MTH\$SQRT_R2	00000000	RG	01
POS	00000005	R	01
SQRTX	00000059	R	01
ZERO_NEG	0000005A	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	0000006E (110.)	01 (1.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOWRT	NOVEC	LONG			

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.07	00:00:01.65
Command processing	127	00:00:00.69	00:00:03.66
Pass 1	79	00:00:00.78	00:00:04.39
Symbol table sort	0	00:00:00.00	00:00:00.00
Pass 2	61	00:00:00.71	00:00:02.94
Symbol table output	2	00:00:00.02	00:00:00.02
Psect synopsis output	3	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	306	00:00:02.29	00:00:12.68

The working set limit was 900 pages.
3664 bytes (8 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 14 non-local and 0 local symbols.
335 source lines were read in Pass 1, producing 8 object records in Pass 2.
1 page of virtual memory was used to define 1 macro.

+-----+
! 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.

MTH\$SQRT2
VAX-11 Macro Run Statistics

; Floating Point Square Root routine^{G 1}

16-SEP-1984 01:51:35 VAX/VMS Macro V04-00 Page 9
6-SEP-1984 11:27:15 [MTHRTL.SRC]MTH\$SQRT2.MAR;1 (5)

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LI\$:MTH\$SQRT2/OBJ=OBJ\$:MTH\$SQRT2 MSRC\$:MTHJACKET/UPDATE=(ENH\$:MTHJACKET)+MS

MT
1-

0263

AH-BT13A-SE
 VAX/VMS V4.

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

0264 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY