


```

BBBBBBBB      AAAAAA      SSSSSSSS      PPPPPPPP      000000      WW      WW      DDDDDDDD      DDDDDDDD
BBBBBBBB      AAAAAA      SSSSSSSS      PPPPPPPP      000000      WW      WW      DDDDDDDD      DDDDDDDD
BB      BB      AA      AA      SS      PP      PP      00      00      WW      WW      DD      DD      DD      DD
BB      BB      AA      AA      SS      PP      PP      00      00      WW      WW      DD      DD      DD      DD
BB      BB      AA      AA      SS      PP      PP      00      00      WW      WW      DD      DD      DD      DD
BB      BB      AA      AA      SS      PP      PP      00      00      WW      WW      DD      DD      DD      DD
BBBBBBBB      AA      AA      SSSSSS      PPPPPPPP      00      00      WW      WW      DD      DD      DD      DD
BBBBBBBB      AA      AA      SSSSSS      PPPPPPPP      00      00      WW      WW      DD      DD      DD      DD
BB      BB      AAAAAAAAAA      SS      PP      00      00      WW      WW      DD      DD      DD      DD
BB      BB      AAAAAAAAAA      SS      PP      00      00      WW      WW      DD      DD      DD      DD
BB      BB      AA      AA      SS      PP      00      00      WWWWWW      WWWWWW      DD      DD      DD      DD
BB      BB      AA      AA      SS      PP      00      00      WWWWWW      WWWWWW      DD      DD      DD      DD
BBBBBBBB      AA      AA      SSSSSSSS      PP      000000      WW      WW      DDDDDDDD      DDDDDDDD
BBBBBBBB      AA      AA      SSSSSSSS      PP      000000      WW      WW      DDDDDDDD      DDDDDDDD

```

```

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

```

BASSPOWDD
Table of contents

: BASIC double ** double routine J 9

15-SEP-1984 23:58:37 VAX/VMS Macro V04-00

Page 0

(2) 54
(3) 91

DECLARATIONS
BASSPOWDD - BASIC double ** double

```
0000 1 .TITLE BAS$POWDD ; BASIC double ** double routine
0000 2 .IDENT /1-006/ ; File: BASPOWDD.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: Basic Support Library
0000 31 :
0000 32 : ABSTRACT:
0000 33 :
0000 34 : This module contains entry points to support exponentiation
0000 35 : (** or ^) in BASIC-PLUS-2 for DOUBLE ** DOUBLE.
0000 36 :
0000 37 : ENVIRONMENT: User Mode, AST Reentrant
0000 38 :
0000 39 :--
0000 40 : AUTHOR: R. Will , CREATION DATE: 22-NOV-78
0000 41 :
0000 42 : MODIFIED BY:
0000 43 :
0000 44 : R. Will, : VERSION 01
0000 45 : 1-01 - Original
0000 46 : 1-02 - Fix comments, make JMP not BRW. RW 5-DEC-78
0000 47 : 1-003 - Add "" to the PSECT directive. JBS 22-DEC-78
0000 48 : 1-004 - Redo case analysis for base leg 0 for compatability
0000 49 : with the PDP-11. JBS 24-APR-1979
0000 50 : 1-005 - Change shared external references to G^ RNH 25-Sep-81
0000 51 : 1-006 - Call OTSSPOWDJ if base > 0 and exponent is integer. MDL 28-Dec-1982
0000 52 :
```

```
0000 54 .SBTTL DECLARATIONS
0000 55 :
0000 56 : INCLUDE FILES:
0000 57 :
0000 58 :
0000 59 :
0000 60 : EXTERNAL DECLARATIONS:
0000 61 :
0000 62 .DSABL GBL ; Prevent undeclared
0000 63 ; symbols from being
0000 64 ; automatically global.
0000 65
0000 66 .EXTRN OTSSPOWDD ; OTSS$ double ** double exponentiation
0000 67 .EXTRN OTSSPOWDJ ; OTSS$ double ** int exponentiation
0000 68 .EXTRN BASSK_DIVBY_ZER ; Divide by Zero
0000 69 .EXTRN BASSK_ILLARGLOG ; Illegal argument in LOG
0000 70 .EXTRN BASS$STOP ; Error reporting routine
0000 71
0000 72 :
0000 73 : MACROS:
0000 74 :
0000 75 :
0000 76 :
0000 77 : EQUATED SYMBOLS:
0000 78 :
0000 79 :
0000 80 :
0000 81 : OWN STORAGE:
0000 82 :
0000 83 :
0000 84 :
0000 85 : PSECT DECLARATIONS:
0000 86 :
00000000 87 .PSECT _BAS$CODE PIC, USR, CON, REL, LCL, SHR, -
0000 88 EXE, RD, NOWRT, LONG
0000 89
```

```

0000 91      .SBTTL  BAS$POWDD  - BASIC double ** double
0000 92      : **
0000 93      : FUNCTIONAL DESCRIPTION:
0000 94      :
0000 95      : This routine takes BASE ** EXP, using the following table
0000 96      : for unusual cases:
0000 97      :
0000 98      : BASE > 0, EXP not integer      Call OTSS$POWDD, normal case.
0000 99      : BASE > 0, EXP integer        Call OTSS$POWDJ
0000 100     : BASE = 0, EXP > 0           Return 0.0.
0000 101     : BASE = 0, EXP = 0         Return 1.0.
0000 102     : BASE = 0, EXP < 0       Error: divide by zero
0000 103     : BASE < 0, EXP even integer Call OTSS$POWDJ with -BASE
0000 104     : BASE < 0, EXP odd integer  Call OTSS$POWDJ with -BASE, negate result
0000 105     : BASE < 0, EXP not integer  Error: illegal argument in LOG.
0000 106     :
0000 107     : CALLING SEQUENCE:
0000 108     :
0000 109     : CALL result.wd.v = BAS$POWDD (base.rd.v, exponent.rd.v)
0000 110     :
0000 111     : INPUT PARAMETERS:
0000 112     :
00000004 0000 113     : base = 4
0000000C 0000 114     : exponent = 12
0000 115     :
0000 116     : IMPLICIT INPUTS:
0000 117     :
0000 118     : NONE
0000 119     :
0000 120     : OUTPUT PARAMETERS:
0000 121     :
0000 122     : NONE
0000 123     :
0000 124     : IMPLICIT OUTPUTS:
0000 125     :
0000 126     : NONE
0000 127     :
0000 128     : FUNCTION VALUE:
0000 129     : COMPLETION CODES:
0000 130     :
0000 131     : double result of exponentiation
0000 132     :
0000 133     : SIDE EFFECTS:
0000 134     :
0000 135     : Will signal Divide By Zero or Illegal argument in LOG if its
0000 136     : arguments are bad, and OTSS$POWDD and OTSS$POWDJ may also signal.
0000 137     :
0000 138     : --
0000 139     :
0000' 0000 140 BAS$POWDD:: .MASK OTSS$POWDD      ; Entry point
0002 141      ; Since this routine uses no
0002 142      ; registers and usually transfers
0002 143      ; control to OTSS$POWDD, we copy
0002 144      ; its register save mask and then
0002 145      ; JMP past its save mask and only
0002 146      ; save the registers once
04 AC 73 0002 147      TSTD  base(AP)      ; Test base relationship to zero

```

```

21 15 0005 148 BLEQ 1$ ; If base leq 0, do case analysis
      0007 149 :+
      0007 150 : Come here if the base is greater than zero. If the exponent is an
      0007 151 : integer, then we can call OTSS$POWDJ.
      0007 152 :-
50 50 08 00 0C AC 74 0007 153 EMODD exponent(AP), #0, #1, R0, R0
      12 12 000E 154 BNEQ 7$ ; Branch if exponent is not integer
      0010 155
      50 0C AC 6A 0010 156 CVTDL exponent(AP), R0 ; Convert exponent to integer
      DD 0014 157 PUSHL R0 ; stack (integer) exponent
      7E 04 AC 70 0016 158 MOVD base(AP), -(SP) ; stack base
00000000'GF 03 FB 001A 159 CALLS #3, G^OTSS$POWDJ ; Call integer power routines
      04 0021 160 RET ; return result
      0022 161 :+
      0022 162 : Come here if the base is greater than zero and the exponent is not
      0022 163 : an integer. This is the general case.
      0022 164 :-
      00000002'GF 17 0022 165 7$: JMP G^OTSS$POWDD+2 ; Transfer control to the OTSS
      0028 166 ; routine to do exponentiation
      0028 167 :+
      0028 168 : Come here if the base is less than or equal to zero. We must filter
      0028 169 : several special cases, as described above.
      0028 170 :-
50 50 08 00 0C 2E 13 0028 171 1$: BEQL 4$ ; Branch if base = 0
      AC 74 002A 172 EMODD exponent(AP), #0, #1, R0, R0
      1A 12 0031 173 BNEQ 3$ ; Branch if exponent is not integer
      0033 174 :+
      0033 175 : The base is less than zero and the exponent is an integer.
      0033 176 : BASIC defines this as working the same way as if an integer was
      0033 177 : in the expression (making a double variable which happens to
      0033 178 : contain an integer value equivalent to an integer variable).
      0033 179 :-
      50 0C AC 6A 0033 180 CVTDL exponent(AP), R0 ; Convert exponent to integer
      DD 0037 181 PUSHL R0 ; Save for even/odd test
      DD 0039 182 PUSHL R0 ; Stack as parameter to OTSS$POWDJ
      7E 04 AC 72 003B 183 MNEGD base(AP), -(SP) ; Stack -base also
00000000'GF 03 FB 003F 184 CALLS #3, G^OTSS$POWDJ ; Call integer power routines
      03 8E E9 0046 185 BLBC (SP)+, 2$ ; Branch if exponent even
      50 50 72 0049 186 MNEGD R0, R0 ; Exponent odd, negate the result
      04 004C 187 2$: RET ; and return with it.
      004D 188 :+
      004D 189 : Come here if the base is less than zero but the exponent is not
      004D 190 : an integer. BASIC defines this as an error.
      004D 191 :-
      7E 0C 8F 9A 004D 192 3$: MOVZBL #BAS$K_ILLARGLOG, -(SP) ; Illegal Argument in LOG
00000000'GF 01 FB 0051 193 CALLS #1, G^BAS$$STOP ; Never return.
      0058 194 :+
      0058 195 : Come here if the base is equal to zero. The value we return depends
      0058 196 : upon the sign of the exponent.
      0058 197 :-
      0C AC 73 0058 198 4$: TSTD exponent(AP) ; Test the exponent against zero
      09 19 005B 199 BLSS 6$ ; Branch if exponent lss 0
      03 13 005D 200 BEQL 5$ ; Branch if exponent is 0
      005F 201 :+
      005F 202 : Come here if the base is zero and the exponent is greater than zero.
      005F 203 : BASIC defines this as 0.0.
      005F 204 :-

```

```
50 7C 005F 205          CLRD   R0          ; R0, R1 = 0.0
    04 0061 206          RET           ; Return to caller
      0062 207          ;+
      0062 208          ; Come here if the base is zero and the exponent is zero. BASIC defines
      0062 209          ; this as 1.0.
      0062 210          ; -
50 08 70 0062 211 5$:   MOVD   #1, R0      ; R0, R1 = 1.0
    04 0065 212          RET           ; Return to caller.
      0066 213          ;+
      0066 214          ; Come here if the base is zero and the exponent is less than zero.
      0066 215          ; BASIC defines this as an error.
      0066 216          ; -
7E 00'8F 9A 0066 217 6$:   MOVZBL #BAS$K_DIVBY_ZER, -(SP) ; Divide by zero
00000000'GF 01 FB 006A 218   CALLS  #1, G^BAS$$STOP ; Report error, never return.
      0071 219          ;
      0071 220          .END
```


BAS\$POWDD ; BASIC double ** double routine C 10
 Symbol table

```

BAS$STOP ***** X 00
BAS$K_DIVBY_ZER ***** X 00
BAS$K_I_L_LARGLOG ***** Y 00
BAS$POWDD 00000000 RG 01
BASE = 00000004
EXPONENT = 0000000C
OTS$POWDD ***** X 00
OTS$POWDJ ***** X 00
  
```

 ! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
_BAS\$CODE	00000071 (113.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

 ! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.08	00:00:00.27
Command processing	119	00:00:00.45	00:00:01.67
Pass 1	74	00:00:00.54	00:00:01.65
Symbol table sort	0	00:00:00.00	00:00:00.00
Pass 2	53	00:00:00.43	00:00:01.46
Symbol table output	2	00:00:00.02	00:00:00.01
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	284	00:00:01.54	00:00:05.09

The working set limit was 900 pages.
 2598 bytes (6 pages) of virtual memory were used to buffer the intermediate code.
 There were 10 pages of symbol table space allocated to hold 8 non-local and 7 local symbols.
 220 source lines were read in Pass 1, producing 8 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\$:BASPOWDD/OBJ=OBJ\$:BASPOWDD MSRC\$:BASPOWDD/UPDATE-(ENH\$:BASPOWDD)

