

BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL

```
BBBBBBBBB      AAAAAA      SSSSSSSS      PPPPPPPP      000000      WW      WW      IIIIII      IIIIII
BBBBBBBBB      AAAAAA      SSSSSSSS      PPPPPPPP      000000      WW      WW      IIIIII      IIIIII
BB      BB      AA      AA      SS      PP      PP      00      00      WW      WW      II      II
BB      BB      AA      AA      SS      PP      PP      00      00      WW      WW      II      II
BB      BB      AA      AA      SS      PP      PP      00      00      WW      WW      II      II
BBBBBBBBB      AA      AA      SSSSSS      PPPPPPPP      00      00      WW      WW      II      II
BBBBBBBBB      AA      AA      SSSSSS      PPPPPPPP      00      00      WW      WW      II      II
BB      BB      AAAAAAAAAA      SS      PP      00      00      WW      WW      II      II
BB      BB      AAAAAAAAAA      SS      PP      00      00      WW      WW      II      II
BB      BB      AA      AA      SS      PP      00      00      WWW      WWW      II      II
BB      BB      AA      AA      SS      PP      00      00      WWW      WWW      II      II
BBBBBBBBB      AA      AA      SSSSSSSS      PP      000000      WW      WW      IIIIII      IIIIII
BBBBBBBBB      AA      AA      SSSSSSSS      PP      000000      WW      WW      IIIIII      IIIIII
```

```
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
```


(2) 58
(3) 94
(4) 211

DECLARATIONS
BASS\$POWII - BASIC word ** word
BASS\$POWII_NIV

```
0000 1 .TITLE BASSPOWII
0000 2 .IDENT /1-006/
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 WORD ** WORD.
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, change BRW to JMP. RW 7-Dec-78
0000 47 : 1-003 - Add "" to the PSECT directive. JBS 22-DEC-78
0000 48 : 1-004 - Redo the case analysis of the BASE leg 0 case for
0000 49 : compatability with the PDP-11. JBS 24-APR-1979
0000 50 : 1-005 - Force references to shared code to use general-addressing
0000 51 : mode. RKR 28-SEP-1981
0000 52 : 1-006 - Check the flags word in the BASIC frame for the setting of
0000 53 : the integer overflow bit and use that value as the setting
0000 54 : (or clearing) of the PSW. Also added entry point
0000 55 : BASS$POWII NIV to ensure that the IV bit in the PSL is
0000 56 : cleared before giving control to OTS$POWII. LB 15-May-1982
```



```

0000 58      .SBTTL  DECLARATIONS
0000 59      :
0000 60      : INCLUDE FILES:
0000 61      :
0000 62      :
0000 63      :
0000 64      : EXTERNAL DECLARATIONS:
0000 65      :
0000 66      : .DSABL  GBL                      ; Prevent undeclared
0000 67      :                                     ; symbols from being
0000 68      :                                     ; automatically global.
0000 69      :
0000 70      : .EXTRN  OT$POWII                    ; OT$ word ** word exponentation
0000 71      : .EXTRN  BAS$K_DIVBY_ZER              ; Divide by Zero
0000 72      : .EXTRN  BAS$$STOP                    ; Error reporting routine
0000 73      : .EXTRN  BAS$HANDLER                  ; BASIC condition handler
0000 74      :
0000 75      :
0000 76      : MACROS:
0000 77      :
0000 78      :
0000 79      :
0000 80      : EQUATED SYMBOLS:
0000 81      :
0000 82      :
0000 83      :
0000 84      : OWN STORAGE:
0000 85      :
0000 86      :
0000 87      :
0000 88      : PSECT DECLARATIONS:
0000 89      :
00000000 90      : .PSECT _BAS$CODE PIC, USR, CON, REL, LCL, SHR, -
0000 91      : EXE, RD, NOWRT, LONG
0000 92

```

```
0000 94 .SBTTL BASS$POWII - BASIC word ** word
0000 95 ;++
0000 96 ; FUNCTIONAL DESCRIPTION:
0000 97 ;
0000 98 ; This routine takes BASE ** EXP, using the following table
0000 99 ; for unusual cases:
0000 100 ;
0000 101 ; BASE > 0 Call OTSS$POWII, normal case.
0000 102 ; BASE = 0, EXP > 0 Return 0.
0000 103 ; BASE = 0, EXP = 0 Return 1.
0000 104 ; BASE = 0, EXP < 0 Error: divide by zero
0000 105 ; BASE < 0, EXP even Call OTSS$POWII with -BASE
0000 106 ; BASE < 0, EXP odd Call OTSS$POWII with -BASE, negate result
0000 107 ;
0000 108 ; CALLING SEQUENCE:
0000 109 ;
0000 110 ; CALL result.ww.v = BASS$POWII (base.rw.v, exponent.rw.v)
0000 111 ;
0000 112 ; INPUT PARAMETERS:
0000 113 ;
0000 114 ; base = 4
0000 115 ; exponent = 8
0000 116 ;
0000 117 ; IMPLICIT INPUTS:
0000 118 ;
0000 119 ; NONE
0000 120 ;
0000 121 ; OUTPUT PARAMETERS:
0000 122 ;
0000 123 ; NONE
0000 124 ;
0000 125 ; IMPLICIT OUTPUTS:
0000 126 ;
0000 127 ; NONE
0000 128 ;
0000 129 ; FUNCTION VALUE:
0000 130 ; COMPLETION CODES:
0000 131 ;
0000 132 ; word result of exponentiation
0000 133 ;
0000 134 ; SIDE EFFECTS:
0000 135 ;
0000 136 ; Will signal Divide By Zero if its arguments are bad,
0000 137 ; and OTSS$POWII may also signal.
0000 138 ;
0000 139 ;--
0000 140 ;
0000' 0000 141 BASS$POWII:: .MASK OTSS$POWII ; Entry point
0002 142 ; Since this routine uses no
0002 143 ; registers and usually transfers
0002 144 ; control to OTSS$POWII, we copy
0002 145 ; its register save mask and then
0002 146 ; JMP past its save mask and only
0002 147 ; save the registers once
0002 148 ;
0002 149 ;+
0002 150 ; On a call to BASS$POWII, the flags word contained in the BASIC frame
```



```
0002 151 ; defines whether integer overflow should or should not be enabled.
0002 152 ; The value of the flag should dictate the setting within the PSL.
0002 153 ;--
51 52 0C AD D0 0002 154 ; MOVL 12(FP),R2 ; Fetch the saved frame pointer
00000000'GF DE 0006 155 ; MOVAL G^BAS$HANDLER,R1 ; Fetch addr of BASIC handler
51 51 62 D1 000D 156 ; CMPL 0(R2),R1 ; Check if this is a BASIC frame
11 12 0010 157 ; BNEQ 9$ ; Branch if not a BASIC frame
52 52 E6 A2 B0 0012 158 ; MOVW -26(R2),R2 ; Fetch flags word from BASIC frame
52 F7FF 8F AA 0016 159 ; BICW #^XF7FF,R2 ; Clear all but IV bit
52 52 B5 001B 160 ; TSTW R2 ; Check if integer overflow is set
02 13 001D 161 ; BEQL 8$ ; Branch if clear
02 11 001F 162 ; BRB 9$ ; Continue as usual
20 B9 0021 163 8$: BICPSW #^X20 ; Clear integer overflow in PSW
04 AC B5 0023 164 9$: TSTW base(AP) ; Test base relationship to zero
06 15 0026 165 ; BLEQ 1$ ; If base leq 0, do case analysis
00000002'GF 17 0028 166 ; JMP G^OTSS$POWII+2 ; Transfer control to the OTSS
002E 167 ; routine to do exponentiation
002E 168 ;+
002E 169 ; Come here if the base is less than or equal to zero. We must filter
002E 170 ; several special cases, as described above.
002E 171 ;--
7E 27 13 002E 172 1$: BEQL 4$ ; Branch if base = 0
50 08 AC 32 0030 173 ; CVTWL exponent(AP), -(SP) ; Stack EXP as parameter to OTSS$POWII
50 04 AC 32 0034 174 ; CVTWL base(AP), R0 ; Get BASE as a longword
7E 50 CE 0038 175 ; MNEGL R0, -(SP) ; Stack -BASE as param to OTSS$POWII
52 B5 003B 176 ; TSTW R2 ; Check if IV is set
09 12 003D 177 ; BNEQ 7$ ; Do the regular CALL to OTSS$POWII
00000070'GF 02 FB 003F 178 ; CALLS #2, G^BAS$POWII_NIV ; Clear IV before JMPing to OTSS$POWII
07 11 0046 179 ; BRB 10$
00000000'GF 02 FB 0048 180 7$: CALLS #2, G^OTSS$POWII ; Call integer power routines
03 08 AC E9 004F 181 10$: BLBC exponent(AP), 2$ ; Branch if exponent even
50 50 AE 0053 182 ; MNEGW R0, R0 ; Exponent odd, negate the result
04 04 0056 183 2$: RET ; and return with it.
0057 184 ;+
0057 185 ; Come here if the base is equal to zero. The value we return depends
0057 186 ; upon the sign of the exponent.
0057 187 ;--
08 AC B5 0057 188 4$: TSTW exponent(AP) ; Test the exponent against zero
09 19 005A 189 ; BLSS 6$ ; Branch if exponent lss 0
03 13 005C 190 ; BEQL 5$ ; Branch if exponent is 0
005E 191 ;+
005E 192 ; Come here if the base is zero and the exponent is greater than zero.
005E 193 ; BASIC defines this as 0.
005E 194 ;--
50 B4 005E 195 ; CLRW R0 ; R0 = 0
04 04 0060 196 ; RET ; Return to caller
0061 197 ;+
0061 198 ; Come here if the base is zero and the exponent is zero. BASIC defines
0061 199 ; this as 1.
50 01 B0 0061 200 ;--
04 04 0064 201 5$: MOVW #1, R0 ; R0 = 1
0065 202 ; RET ; Return to caller.
0065 203 ;+
0065 204 ; Come here if the base is zero and the exponent is less than zero.
0065 205 ; BASIC defines this as an error.
0065 206 ;--
7E 00'8F 9A 0065 207 6$: MOVZBL #BAS$K_DIVBY_ZER, -(SP) ; Divide by zero
```

BAS\$POWII
1-006

: BASIC integer ** integer
BAS\$POWII - BASIC word ** word

M 13

16-SEP-1984 00:00:09 VAX/VMS Macro V04-00
6-SEP-1984 10:34:25 [BASRTL.SRC]BASPOWII.MAR;1

Page 5
(3)

00000000'GF 01 FB 0069 208 CALLS #1, G^BAS\$\$STOP ; Report error, never return.
0070 209 ;


```

0070 211 .SBTTL BASS$POWII_NIV
0070 212
0070 213 :++
0070 214 : Functional Description:
0070 215 :
0070 216 : This routine is an internal entry point, called only by BASS$POWII
0070 217 : whose sole purpose in this world is to turn off the integer
0070 218 : overflow bit in the PSL for the case where the base is less than
0070 219 : zero, and where the flags bit in the BASIC frame indicate to turn
0070 220 : off integer overflow.
0070 221 :
0070 222 : Calling Sequence:
0070 223 :
0070 224 : CALL BASS$POWII_NIV (base.rw.v, exponent.rw.v)
0070 225 :
0070 226 : Input Parameters:
0070 227 :
0070 228 : None
0070 229 :
0070 230 : Output Parameters:
0070 231 :
0070 232 : None
0070 233 :
0070 234 : Side Effects:
0070 235 :
0070 236 : OTS$POWII may signal
0070 237 :--
0070 238
0070 239 BASS$POWII_NIV::
0070 240 .MASK OTS$POWII ; Entry point
0072 241 BICPSW #^X20 ; Clear IV bit in PSL
0074 242 JMP G^OTS$POWII+2 ; Transfer control to the OTS$
007A 243 ; routine to do exponentiation
007A 244
007A 245 .END

```

00000002'GF 20 0000' B9 17

BAS\$POWII ; BASIC integer ** integer
Symbol table

B 14

16-SEP-1984 00:00:09 VAX/VMS Macro V04-00
6-SEP-1984 10:34:25 [BASRTL.SRC]BASPOWII.MAR;1

Page 7
(4)

BAS\$\$POWII_NIV 00000070 RG 01
BAS\$\$STOP ***** X 00
BAS\$HANDLER ***** X 00
BAS\$K_DIVBY_ZER ***** X 00
BAS\$POWII 00000000 RG 01
BASE = 00000004
EXPONENT = 00000008
OT\$POWII ***** 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	0000007A (122.)	01 (1.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOWRT	NOVEC	LONG					

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.08	00:00:00.60
Command processing	108	00:00:00.50	00:00:02.66
Pass 1	69	00:00:00.54	00:00:01.15
Symbol table sort	0	00:00:00.00	00:00:00.00
Pass 2	55	00:00:00.48	00:00:01.28
Symbol table output	2	00:00:00.02	00:00:00.01
Psect synopsis output	2	00:00:00.02	00:00:00.06
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	270	00:00:01.65	00:00:05.77

The working set limit was 900 pages.
2846 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 9 local symbols.
245 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\$:BASPOWII/OBJ=OBJ\$:BASPOWII MSRC\$:BASPOWII/UPDATE=(ENH\$:BASPOWII)

0029 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BASOPEN
LIS

BASPOWJ
LIS

BASPOS
LIS

BASPOWJ
LIS

BASOPENDE
LIS

BASPOWGG
LIS

BASPOWH
LIS

BASPOWRJ
LIS

BASPOWII
LIS

BASPURIOB
LIS

BASPOWDO
LIS

BASOPENZE
LIS

BASPOWR
LIS

BASPOWJ
LIS

BASPOWR
LIS

BASPOWH
LIS

BASPOWR
LIS