

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

```

PPPPPPPPPPPP      AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTTTT      LLL
PPPPPPPPPPPP      AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTTTT      LLL
PPPPPPPPPPPP      AAAAAAAAAA      SSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTTTT      LLL
PPP                PPP      AAA                AAA      SSS                SSS                TTT                TTT                LLL
PPP                PPP      AAA                AAA      SSS                SSS                TTT                TTT                LLL
PPP                PPP      AAA                AAA      SSS                SSS                TTT                TTT                LLL
PPP                PPP      AAA                AAA      SSS                SSS                TTT                TTT                LLL
PPP                PPP      AAA                AAA      SSS                SSS                TTT                TTT                LLL
PPP                PPP      AAA                AAA      SSS                SSS                TTT                TTT                LLL
PPPPPPPPPPPP      AAA                AAA                SSSSSSSSSS      RRRRRRRRRRRR      TTT                TTT                LLL
PPPPPPPPPPPP      AAA                AAA                SSSSSSSSSS      RRRRRRRRRRRR      TTT                TTT                LLL
PPPPPPPPPPPP      AAA                AAA                SSSSSSSSSS      RRRRRRRRRRRR      TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL
PPP                AAA                AAA                SSS                RRR                RRR                TTT                TTT                LLL

```

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

PAS

```

PPPPPPPP      AAAAAA      SSSSSSSS      000000      CCCCCCCC      TTTTTTTTTT
PPPPPPPP      AAAAAA      SSSSSSSS      000000      CCCCCCCC      TTTTTTTTTT
PP      PP      AA      AA      SS      00      00      CC      TT
PP      PP      AA      AA      SS      00      00      CC      TT
PP      PP      AA      AA      SS      00      00      CC      TT
PP      PP      AA      AA      SS      00      00      CC      TT
PPPPPPPP      AA      AA      SSSSSS      00      00      CC      TT
PPPPPPPP      AA      AA      SSSSSS      00      00      CC      TT
PP      AAAAAAAAAA      SS      00      00      CC      TT
PP      AAAAAAAAAA      SS      00      00      CC      TT
PP      AA      AA      SS      00      00      CC      TT
PP      AA      AA      SS      00      00      CC      TT
PP      AA      AA      SSSSSSSS      000000      CCCCCCCC      TT
PP      AA      AA      SSSSSSSS      000000      CCCCCCCC      TT

```

```

....
....
....
....

```

```

LL      111111      SSSSSSSS
LL      111111      SSSSSSSS
LL      11      SS
LL      11      SS
LL      11      SS
LL      11      SS
LL      11      SSSSSS
LL      11      SSSSSS
LL      11      SS
LL      11      SS
LL      11      SS
LL      11      SS
LLLLLLLLLLLL 111111      SSSSSSSS
LLLLLLLLLLLL 111111      SSSSSSSS

```

```

1 0001 0 MODULE PASSOCT ( %TITLE 'Convert value in base 8 to string'
2 0002 0 IDENT = '1-001' ! File: PASSOCT.B32 Edit: SBL1001
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
10 0010 1 * ALL RIGHTS RESERVED. *
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
17 0017 1 * TRANSFERRED. *
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
21 0021 1 * CORPORATION. *
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1 ++
31 0031 1 FACILITY: Pascal Language Support
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1
35 0035 1 This module contains PASSOCT which implements the
36 0036 1 VAX-11 Pascal OCT procedure.
37 0037 1
38 0038 1 ENVIRONMENT: User mode - AST reentrant
39 0039 1
40 0040 1 AUTHOR: Steven B. Lionel, CREATION DATE: 1-April-1981
41 0041 1
42 0042 1 MODIFIED BY:
43 0043 1
44 0044 1 1-001 - Original. SBL 1-April-1981
45 0045 1 --
46 0046 1

```

```

48 0047 1 %SBTTL 'Declarations'
49 0048 1
50 0049 1 : PROLOGUE DEFINITIONS:
51 0050 1
52 0051 1
53 0052 1 REQUIRE 'RTLIN:PASPROLOG';           ! Externals, Linkages, PSECTs, structures
54 0116 1
55 0117 1
56 0118 1 : TABLE OF CONTENTS:
57 0119 1
58 0120 1
59 0121 1 FORWARD ROUTINE
60 0122 1     PASSOCT: NOVALUE;                 ! Convert value in base 8
61 0123 1
62 0124 1
63 0125 1 : MACROS:
64 0126 1
65 0127 1     NONE
66 0128 1
67 0129 1 : EQUATED SYMBOLS:
68 0130 1
69 0131 1     NONE
70 0132 1
71 0133 1 : FIELDS:
72 0134 1
73 0135 1     NONE
74 0136 1
75 0137 1 : OWN STORAGE:
76 0138 1
77 0139 1     NONE
78 0140 1
79 0141 1
80 0142 1 !! If this is for a V2 system, redefine OTSSCVT_L_TO as PASSCVT_L_TO.
81 L 0143 1 %IF %VARIANT
82 U 0144 1 %THEN
83 U 0145 1 UNDECLARE
84 U 0146 1     OTSSCVT_L_TO;
85 U 0147 1 EXTERNAL ROUTINE
86 U 0148 1     PASSCVT_L_TO;
87 U 0149 1 BIND ROUTINE
88 U 0150 1     OTSSCVT_L_TO = PASSCVT_L_TO;
89 0151 1 %FI

```

```

91 0152 1 %SBTTL 'PASSOCT - Convert value in base 8 to string'
92 0153 1 GLOBAL ROUTINE PASSOCT (
93 0154 1     RESULT: REF VECTOR [, BYTE],
94 0155 1     TOTAL_WIDTH: WORD SIGNED,
95 0156 1     NBITS,
96 0157 1     VALUE,
97 0158 1     MIN_DIGITS: SIGNED
98 0159 1 ): NOVALUE =
99 0160 1
100 0161 1 ++
101 0162 1 FUNCTIONAL DESCRIPTION:
102 0163 1
103 0164 1     This procedure implements the VAX-11 Pascal OCT function. It
104 0165 1     converts a value to an ASCII representation in base 8 and stores
105 0166 1     that result in a string.
106 0167 1
107 0168 1 CALLING SEQUENCE:
108 0169 1
109 0170 1     CALL PASSOCT (RESULT.wt.r, TOTAL_WIDTH.rw.v, NBITS.rl.v, VALUE.rz.r
110 0171 1     [, MIN_DIGITS.rl.v])
111 0172 1
112 0173 1 FORMAL PARAMETERS:
113 0174 1
114 0175 1     RESULT           - The string into which the result will be placed.
115 0176 1
116 0177 1     TOTAL_WIDTH      - Total field width.
117 0178 1
118 0179 1     NBITS            - The size of VALUE in bits.
119 0180 1
120 0181 1     VALUE           - The address of the value to write.
121 0182 1
122 0183 1     MIN_DIGITS       - Optional. The minimum number of digits to appear
123 0184 1     in the result. Defaults to the minimum necessary to
124 0185 1     represent every bit of the value.
125 0186 1
126 0187 1 IMPLICIT INPUTS:
127 0188 1
128 0189 1     NONE
129 0190 1
130 0191 1 IMPLICIT OUTPUTS:
131 0192 1
132 0193 1     NONE
133 0194 1
134 0195 1 ROUTINE VALUE:
135 0196 1
136 0197 1     NONE
137 0198 1
138 0199 1 SIDE EFFECTS:
139 0200 1
140 0201 1     NONE
141 0202 1
142 0203 1 SIGNALLED ERRORS:
143 0204 1
144 0205 1     NEGDIGARG - negative "digits" argument to BIN, HEX or OCT is not allowed
145 0206 1
146 0207 1 --
147 0208 1

```

```

: 148 0209 2 BEGIN
: 149 0210 2
: 150 0211 2 LOCAL
: 151 0212 2     ACTUAL_DIGITS,           ! Number of digits actually used
: 152 0213 2     ACTUAL_NBITS,         ! Value size actually used
: 153 0214 2     DESCR: "BLOCK" [8, BYTE]; ! String descriptor
: 154 0215 2
: 155 0216 2 LITERAL
: 156 0217 2     M_SIZE_IN_BITS = %X'04'; ! Flags argument for
: 157 0218 2                                           ! convert routine
: 158 0219 2 BUILTIN
: 159 0220 2     ACTUALCOUNT;
: 160 0221 2
: 161 0222 2 !+
: 162 0223 2 ! Set initial values for conversion.
: 163 0224 2 !-
: 164 0225 2
: 165 0226 2     ACTUAL_NBITS = .NBITS;
: 166 0227 2     ACTUAL_DIGITS = (.ACTUAL_NBITS+2)/3;
: 167 0228 2
: 168 0229 2 !+
: 169 0230 2 ! Create result string descriptor with actual width.
: 170 0231 2 !-
: 171 0232 2
: 172 0233 2     DESCR [DSC$B_CLASS] = DSC$K_CLASS_S;
: 173 0234 2     DESCR [DSC$B_DTYPE] = DSC$K_DTYPE_T;
: 174 0235 2     DESCR [DSC$A_POINTER] = RESULT [0];
: 175 0236 2
: 176 0237 2     IF ACTUALCOUNT () GEQU 5
: 177 0238 2     THEN
: 178 0239 2         BEGIN
: 179 0240 2             ACTUAL_DIGITS = .MIN_DIGITS;
: 180 0241 2             IF .ACTUAL_DIGITS LSS 0
: 181 0242 2             THEN
: 182 0243 2                 SIGNAL_STOP (PASS$NEGDIGARG); ! Negative "digits" argument to BIN, HEX or OCT is not allow
: 183 0244 2             END;
: 184 0245 2
: 185 0246 2     DESCR [DSC$W_LENGTH] = .TOTAL_WIDTH;
: 186 0247 2     !+
: 187 0248 2     ! Will TOTAL_WIDTH truncate the value?
: 188 0249 2     !+
: 189 0250 2     IF .TOTAL_WIDTH LSSU (.ACTUAL_NBITS+2)/3
: 190 0251 2     THEN
: 191 0252 2         ACTUAL_NBITS = .TOTAL_WIDTH * 3;
: 192 0253 2
: 193 0254 2     IF .ACTUAL_DIGITS GTRU .TOTAL_WIDTH
: 194 0255 2     THEN
: 195 0256 2         ACTUAL_DIGITS = .TOTAL_WIDTH;
: 196 0257 2
: 197 0258 2 !+
: 198 0259 2 ! Do the conversion. We assume it won't fail.
: 199 0260 2 !-
: 200 0261 2
: 201 0262 2     OTSS$CVT_L_TO (.VALUE, DESCR, .ACTUAL_DIGITS, .ACTUAL_NBITS,
: 202 0263 2         M_SIZE_IN_BITS);
: 203 0264 2
: 204 0265 2     RETURN;

```

PASSOCT  
1-001

Convert value in base 8 to string  
PASSOCT - Convert value in base 8 to string

N 9  
16-Sep-1984 01:45:34  
14-Sep-1984 12:51:40

VAX-11 Bliss-32 V4.0-742  
[PASRTL.SRC]PASSOCT.B32;1

Page 5  
(3)

: 205  
: 206

0266 2  
0267 1     END;

! End of routine PASSOCT

.TITLE PASSOCT Convert value in base 8 to string  
.IDENT \1-001\

.EXTRN PASSOCT, PASS\$ \_NEGDIGARG  
.EXTRN OTSS\$CVT\_L\_TO

.PSECT \_PASS\$CODE, NOWRT, SHR, PIC, 2

.ENTRY PASSOCT, Save R2, R3, R4

SUBL2 #8, SP  
MOVL NBITS, ACTUAL\_NBITS  
MOVAB 2(R2), R3  
DIVL2 #3, R3  
MOVL R3, ACTUAL\_DIGITS  
MOVW #270, DESCR+2  
MOVL RESULT, DESCR+4  
CMPB (AP), #5  
BLSSU 1\$  
MOVL MIN\_DIGITS, ACTUAL\_DIGITS  
BGEQ 1\$  
PUSHL #PASS\$ \_NEGDIGARG  
CALLS #1, LIB\$STOP  
CVTWL TOTAL\_WIDTH, R0  
MOVW R0, DESCR  
CML R0, R3  
BGEQU 2\$  
MULL3 #3, R0, ACTUAL\_NBITS  
CML ACTUAL\_DIGITS, R0  
BLEQU 3\$  
MOVL R0, ACTUAL\_DIGITS  
PUSHL #4  
PUSHL ACTUAL\_NBITS  
PUSHL ACTUAL\_DIGITS  
PUSHAB DESCR  
PUSHL VALUE  
CALLS #5, OTSS\$CVT\_L\_TO  
RET

: 0153  
: 0226  
: 0227  
: 0234  
: 0235  
: 0237  
: 0240  
: 0241  
: 0243  
: 0246  
: 0250  
: 0252  
: 0254  
: 0256  
: 0262  
: 0267

001C 00000  
5E 08 C2 00002  
52 OC AC D0 00005  
53 02 A2 9E 00009  
53 03 C6 0000D  
54 53 D0 00010  
02 AE 010E 8F B0 00013  
04 AE 04 AC D0 00019  
05 6C 91 0001E  
13 1F 00021  
54 14 AC D0 00023  
0D 18 00027  
00000000G 00 00000000G 8F DD 00029  
01 FB 0002F  
50 08 AC 32 00036 1\$:  
6E 50 B0 0003A  
53 50 D1 0003D  
04 1E 00040  
52 50 03 C5 00042  
50 54 D1 00046 2\$:  
03 1B 00049  
54 50 D0 0004B  
04 DD 0004E 3\$:  
52 DD 00050  
54 DD 00052  
OC AE 9F 00054  
10 AC DD 00057  
00000000G 00 05 FB 0005A  
04 00061

; Routine Size: 98 bytes,     Routine Base: \_PASS\$CODE + 0000

: 207     0268 1  
: 208     0269 1 !<BLF/PAGE>

PASSOCT  
1-001

Convert value in base 8 to string  
PASSOCT - Convert value in base 8 to string

B 10  
16-Sep-1984 01:45:34  
14-Sep-1984 12:51:40

VAX-11 Bliss-32 V4.0-742  
[PASRTL.SRC]PASOCT.B32;1

: 210 0270 1 END  
: 211 0271 1  
: 212 0272 0 ELUDOM

. End of module PASSOCT

.EXTRN LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
_PASSCODE	98	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(2)

Library Statistics

File	Total	Symbols Loaded	Percent	Pages Mapped	Processing Time
\$_255\$DUA28:[SYSLIB]STARLET.L32;1	9776	6	0	581	00:01.0
\$_255\$DUA28:[PASRTL.OBJ]PASLIB.L32;1	427	3	0	33	00:00.4

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD, INITIAL, OPTIMIZE)/NOTRACE/LIS=LIS\$:PASOCT/OBJ=OBJ\$:PASOCT MSRC\$:PASOCT/UPDATE=(ENH\$:PASOCT)

: Size: 98 code + 0 data bytes  
: Run Time: 00:04.6  
: Elapsed Time: 00:18.0  
: Lines/CPU Min: 3586  
: Lexemes/CPU-Min: 8070  
: Memory Used: 54 pages  
: Compilation Complete



0295 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

The image displays a grid of 144 small terminal window screenshots, arranged in 12 rows and 12 columns. Each window shows a different program name followed by the letters 'LIS', representing various List Processing (LIS) programs. The programs visible include:

- PASPAGE2 LIS
- PASODD LIS
- PASHEX LIS
- PASLINEL LIS
- PASMSGPTR LIS
- PASLIB LIS
- PASLOOKAH LIS
- PASOCT LIS
- PASOPEN2 LIS
- PASPUT LIS
- PASQHAND LIS
- PASLOCATE LIS
- PASMSGTXT LIS

The screenshots are very small and mostly illegible, but they consistently follow the pattern of [Program Name] LIS.