

A large, stylized letter 'A' is formed using the characters 'S' and 'Y'. The left and right vertical strokes are composed of 'S' characters, while the central vertical stroke and the two diagonal strokes are composed of 'Y' characters. The 'A' is symmetrical and has a blocky, digital appearance.

```
EEEEEEEEEE XX      XX      CCCCCCCC EEEEEEEEE PP PPPPPP TTTTTTTTT MM      MM      SSSSSSSS GGGGGGGG
EEEEEEEEEE XX      XX      CCCCCCCC EEEEEEEEE PP PPPPPP TTTTTTTTT MM      MM      SSSSSSSS GGGGGGGG
EE          XX      XX      CC          PP      PP      TT      MMMM  MMMM  SS          GG
EE          XX      XX      CC          PP      PP      TT      MMMM  MMMM  SS          GG
EE          XX      XX      CC          PP      PP      TT      MM   MM   SS          GG
EEEEEEEEEE      XX      XX      CC          PPPPPPP PP      TT      MM   MM   SSSSSS  GG
EEEEEEEEEE      XX      XX      CC          PPPPPPP PP      TT      MM   MM   SSSSSS  GG
EE          XX      XX      CC          PP      PP      TT      MM   MM   SS          GG
EE          XX      XX      CC          PP      PP      TT      MM   MM   SS          GG
EE          XX      XX      CC          PP      PP      TT      MM   MM   SS          GG
EEEEEEEEEE XX      XX      CCCCCCCC EEEEEEEEE PP      PP      TT      MM   MM   SSSSSSSS GGGGGG
EEEEEEEEEE XX      XX      CCCCCCCC EEEEEEEEE PP      PP      TT      MM   MM   SSSSSSSS GGGGGG
```

```
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
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS
```


(1) 41
(1) 84
(1) 179

DECLARATIONS
EXCEPTION MESSAGE FORMAT AND PRINT ROUTINE
UTILITY SUBROUTINES


```
0000 1      .TITLE EXCEPTMSG - EXCEPTION MESSAGE FORMAT AND PRINT ROUTINE
0000 2      .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6
0000 7 *
0000 8 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 9 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 10 * ALL RIGHTS RESERVED.
0000 11 *
0000 12 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 13 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 14 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 15 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 16 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 17 * TRANSFERRED.
0000 18 *
0000 19 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 20 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 21 * CORPORATION.
0000 22 *
0000 23 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 24 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 25 *
0000 26 *****
0000 27
0000 28 D. N. CUTLER 9-NOV-77
0000 29
0000 30 EXCEPTION MESSAGE FORMAT AND PRINT ROUTINE
0000 31
0000 32 Modified by:
0000 33
0000 34 V02-003 LMK0001 Len Kowell 3-Sep-1981
0000 35 Change PSECT names to re-order system image.
0000 36
0000 37 V02-002 LJK0033 Lawrence J. Kenah 2-Jun-1981
0000 38 Add dump of top of stack to output display
0000 39
```



```
0000 41 .SUBTITLE DECLARATIONS
0000 42 :
0000 43 : MACRO LIBRARY CALLS
0000 44 :
0000 45 :
0000 46 $CHFDEF ;DEFINE CONDITION HANDLING ARGVST OFFSETS
0000 47 $FABDEF ;DEFINE FAB OFFSETS
0000 48 $RABDEF ;DEFINE RAB OFFSETS
0000 49 :
0000 50 :
0000 51 : LOCAL SYMBOLS
0000 52 :
0000 53 : ARGUMENT LIST OFFSET DEFINITIONS
0000 54 :
0000 55 :
00000004 0000 56 MESSAGE=4 ;ADDRESS OF ASCII STRING
00000008 0000 57 SIGNAL=8 ;ADDRESS OF SIGNAL ARGUMENT LIST
0000 58 :
0000 59 :
0000 60 : OUTPUT FORMAT BUFFER SIZE
0000 61 :
0000 62 :
00000050 0000 63 BUFSIZ=80 ;
0000 64 :
0000 65 :
0000 66 : LOCAL DATA
0000 67 :
0000 68 :
00000000 0000 69 .PSECT YEXEPAGED2,BYTE
79 6C 72 65 70 6F 72 70 6D 49 20 20 0000 70 HEADNG: .ASCII / Improperly handled condition, / ;
6E 6F 63 20 64 65 6C 64 6E 61 68 20 000C
00 20 2C 6E 6F 69 74 69 64 0018
75 64 20 72 65 74 73 69 67 65 52 09 0021 71 REGMSG: .ASCII / Register dump/ ;
00 70 6D 002D
20 31 52 20 20 00 20 3D 20 30 52 09 0030 72 REGDMP: .ASCII / R0 = /<0>/ R1 = /<0>/ R2 = /<0>/ R3 = /<0><128> ;
20 00 20 3D 20 32 52 20 20 00 20 3D 003C
80 00 20 3D 20 33 52 20 0048
20 35 52 20 20 00 20 3D 20 34 52 09 0050 73 .ASCII / R4 = /<0>/ R5 = /<0>/ R6 = /<0>/ R7 = /<0><128> ;
20 00 20 3D 20 36 52 20 20 00 20 3D 005C
80 00 20 3D 20 37 52 20 0068
20 39 52 20 20 00 20 3D 20 38 52 09 0070 74 .ASCII / R8 = /<0>/ R9 = /<0>/ R10= /<0>/ R11= /<0><128> ;
20 00 20 3D 20 31 52 20 20 00 20 3D 007C
80 00 20 3D 20 31 52 20 0088
20 50 46 20 20 00 20 3D 20 50 41 09 0090 75 .ASCII / AP = /<0>/ FP = /<0>/ SP = /<0>/ PC = /<0><128> ;
20 00 20 3D 20 50 53 20 20 00 20 3D 009C
80 00 20 3D 20 43 50 20 00A8
80 00 20 3D 20 4C 53 50 09 00B0 76 .ASCII / PSL= /<0><128> ;
00 20 09 09 00B8 77 SIGCTL: .ASCII / / ;
00BC 78 SIGCTL_BY 3:
75 67 72 61 20 6C 61 6E 67 69 53 09 00BC 79 .ASCII / /
20 20 20 20 20 20 09 73 74 6E 65 6D 00C3 80 SIGMSG: .ASCII / Signal arguments / Stack contents/
6E 65 74 6E 6F 63 20 6B 63 61 74 53 00CF
00 73 74 00E7
00 20 3D 20 20 20 65 6D 61 4E 09 00EA 81 SIGNAM: .ASCII / Name = / ;SIGNAL NAME
00 20 3D 20 72 65 62 6D 75 4E 09 00F5 82 SIGNUM: .ASCII / Number = / ;SIGNAL ARGUMENTS NUMBER
```



```
0100 84 .SBTTL EXCEPTION MESSAGE FORMAT AND PRINT ROUTINE
0100 85 :+
0100 86 : EXE$EXCMMSG - EXCEPTION MESSAGE FORMAT AND PRINT ROUTINE
0100 87 :
0100 88 : THIS ROUTINE IS CALLED WHEN NO EXCEPTION HANDLER CAN BE FOUND OR ALL HANDLERS
0100 89 : RESIGNAL. ITS FUNCTION IS TO PRINT A SUMMARY OF THE CURRENT PROCESS STATE AND
0100 90 : WHY THE PROCESS RECEIVED THE EXCEPTION.
0100 91 :
0100 92 : INPUTS:
0100 93 :
0100 94 : MESSAGE(AP) = ADDRESS OF MESSAGE SUFFIX STRING.
0100 95 : SIGNAL(AP) = ADDRESS OF SIGNAL ARGUMENT LIST.
0100 96 :
0100 97 : OUTPUTS:
0100 98 :
0100 99 : THE HEADING MESSAGE IS FORMATTED BY ADDING THE SPECIFIED SUFFIX AND
0100 100 : IS THEN WRITTEN TO 'SYS$ERROR'. THIS INFORMATION IS FOLLOWED BY A DUMP
0100 101 : OF THE SIGNAL ARGUMENTS AND GENERAL REGISTERS. WHILE THE OUTPUT IS
0100 102 : BEING WRITTEN TO 'SYS$ERROR' IT IS ALSO WRITTEN TO 'SYS$OUTPUT' IF
0100 103 : THEY ARE DIFFERENT FILES.
0100 104 :-
0100 105
0100 106 .ENTRY EXE$EXCMMSG,*M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
0102 107 BSBW EXE$OPEN MSG ;OPEN MESSAGE FILES AND INITIALIZE RAB'S
0105 108 MOVAB -BUFSIZ(SP),SP ;ALLOCATE SPACE FOR OUTPUT BUFFER
0109 109 MOVL SP,R7 ;SAVE ADDRESS OF OUTPUT BUFFER
010C 110 MOVAB -17*4(SP),SP ;ALLOCATE REGISTER SAVE AREA
0110 111 MOVL SIGNAL(AP),R6 ;GET ADDRESS OF SIGNAL ARGUMENTS
0114 112 MOVL SP,R0 ;SET ADDRESS OF REGISTER SAVE AREA
0117 113 MOVL CHF$M_MCHARGLST(R6),R1 ;GET ADDRESS OF MECHANISM ARRAY
011B 114 MOVQ CHF$M_MCH_SAVRO(R1),(R0)+ ;INSERT ORIGINAL R0 AND R1
011F 115 MOVAB 20(FP),R1 ;GET ADDRESS OF SAVED REGISTERS
0123 116 MOVL #10,R2 ;SET REGISTER COUNT
0126 117 10$: MOVL (R1)+(R0)+ ;INSERT ORIGINAL REGISTERS
0129 118 SOBGTR R2,10$ ;ANY MORE REGISTERS TO SAVE?
012C 119 MOVQ 8(FP),(R0)+ ;INSERT ARGUMENT AND FRAME POINTERS
0130 120 MOVL CHF$M_SIGARGLST(R6),R6 ;GET ADDRESS OF SIGNAL ARRAY
0134 121 MOVL (R6),R5 ;GET NUMBER OF SIGNAL ARGUMENTS
0137 122 MOVAL 4(R6)[R5],R8 ;INSERT STACK POINTER ADDRESS
013C 123 MOVL R8,(R0)+ ;USE R8 AS STACK LIST TRAVERSAL REGISTER
013F 124 MOVL -4(R6)[R5],(R0)+ ;INSERT SAVED PROGRAM COUNTER
0144 125 MOVL (R6)[R5],(R0) ;INSERT SAVED PROCESSOR STATUS
0148 126 MOVAB (R7),RAB$L_RBF(R10) ;SET ADDRESS OF RECORD BUFFER
014C 127 MOVAB (R7),RAB$L_RBF(R11) ;SET ADDRESS OF RECORD BUFFER
0150 128 BSBW PUT ;OUTPUT RECORD
0153 129 MOVAB HEADNG,R1 ;GET ADDRESS OF INITIAL HEADING
0158 130 BSBW INSERT ;INSERT HEADING IN OUTPUT BUFFER
015B 131 MOVL MESSAGE(AP),R1 ;GET ADDRESS OF MESSAGE SUFFIX
015F 132 BSBW PUTI ;INSERT SUFFIX IN OUTPUT BUFFER AND PRINT
0162 133 MOVAB SIGMSG,R1 ;GET ADDRESS OF SIGNAL ARGUMENT MESSAGE
0167 134 BSBW PUTI ;INSERT MESSAGE IN OUTPUT BUFFER AND PRINT
016A 135 SUBL3 #1,(R6),R5 ;CALCULATE NUMBER OF SIGNAL ARGUMENTS - 1
016E 136 SUBL3 #8,R5,-(SP) ;STORE DIFFERENCE ON STACK
0172 137 BLEQU 15$ ;BRANCH IF SIGNAL ARRAY SMALLER
0174 138 MOVL #8,R5 ;STACK DISPLAY IS SMALLER
0177 139 15$: MOVAB SIGNUM,R1 ;GET ADDRESS OF ARGUMENT NUMBER PREFIX
017C 140 BSBW PUTC_BOTH ;WRITE TWO COLUMN ENTRIES
```

5E	BO	AE	9E	0105	108
57	SE	DO	0109	109	
5E	BC	AE	9E	010C	110
56	08	AC	DO	0110	111
50	SE	DO	0114	112	
51	08	A6	DO	0117	113
80	0C	A1	7D	011B	114
51	14	AD	9E	011F	115
52	0A	DO	0123	116	
80	81	DO	0126	117	10\$:
FA	52	F5	0129	118	
80	08	AD	7D	012C	119
56	04	A6	DO	0130	120
55	66	DO	0134	121	
58	04	A645	DE	0137	122
80	80	58	DO	013C	123
80	FC	A645	DO	013F	124
60	6645	DO	0144	125	
28	AA	67	9E	0148	126
28	AB	67	9E	014C	127
00A6	30	0150	128		
51	FEA9	CF	9E	0153	129
0085	30	0158	130		
51	04	AC	DO	015B	131
0093	30	015F	132		
51	FF5D	CF	9E	0162	133
0088	30	0167	134		
55	66	01	C3	016A	135
7E	55	08	C3	016E	136
03	1B	0172	137		
55	08	DO	0174	138	
51	FF7A	CF	9E	0177	139
00AE	30	017C	140		15\$:


```
51 FF67 CF 9E 017F 141 MOVAB SIGNAM,R1 ;GET ADDRESS OF SIGNAL NAME PREFIX
    00A6 30 0184 142 BSBW PUTC_BOTH ;WRITE TWO COLUMN ENTRIES
51 FF2D CF 9E 0187 143 20$: MOVAB SIGCTL,R1 ;GET ADDRESS OF INDENT MESSAGE
    009E 30 018C 144 BSBW PUTC_BOTH ;WRITE TWO COLUMN ENTRIES
    F5 55 F5 018F 145 SOBGTR R5,20$ ;ANY MORE TO CONVERT?
    55 BED0 0192 146 POPL R5 ;RETRIEVE DIFFERENCE
    1E 13 0195 147 BEQL REG_DUMP ;NO DIFFERENCE. GO DUMP REGISTERS
    10 14 0197 148 BGTR 25$ ;SIGNAL ARRAY LARGER. DUMP BALANCE.
51 FF1F CF 9E 0199 149 22$: MOVAB SIGCTL_BY_3,R1 ;DUMP BALANCE OF STACK (UP TO 10 LONGWORDS)
    40 10 019E 150 BSBB INSERT ;INSERTED EXTENDED INDENT INTO OUTPUT RECORD
    0096 30 01A0 151 BSBW PUTC_R8 ;OUTPUT THIS STACK LONGWORD
    F2 55 00 F2 01A3 152 AOBLS #0,R5,22$ ;GO GET NEXT ONE
    0C 11 01A7 153 BRB REG_DUMP ;ALL DONE. DUMP REGISTERS
    9E 01A9 154
51 FF0B CF 9E 01A9 155 25$: MOVAB SIGCTL,R1 ;DUMP BALANCE OF SIGNAL ARRAY
    30 10 01AE 156 BSBB INSERT ;INSERTED INDENT INTO OUTPUT RECORD
    77 10 01B0 157 BSBB PUTC_R6 ;OUTPUT THIS SIGNAL ARRAY ELEMENT
    F4 55 F5 01B2 158 SOBGTR R5,25$ ;GO GET NEXT ONE
    01B5 159
    01B5 160 REG_DUMP:
51 FE66 CF 9E 01B5 161 BSBB PUT ;OUTPUT BLANK LINE
    37 10 01B7 162 MOVAB REGMSG,R1 ;GET ADDRESS OF REGISTER DUMP MESSAGE
    54 05 D0 01BC 163 BSBB PUTI ;INSERT MESSAGE IN OUTPUT BUFFER AND PRINT
55 FE6B CF 9E 01BE 164 MOVL #5,R4 ;SET NUMBER OF LINES TO PRINT
    56 5E D0 01C1 165 MOVAB REGDMP,R5 ;SET ADDRESS OF REGISTER DUMP CONTROL STRING
    50 85 98 01C6 166 MOVL SP,R6 ;SET ADDRESS OF REGISTER SAVE AREA
    04 15 01CC 167 30$: CVTBL (R5)+,R0 ;GET NEXT BYTE FROM CONTROL STRING
    19 10 01CE 168 BLEQ 40$ ;IF LEQ CONTROL BYTE
    F7 11 01D0 169 BSBB PUTB ;PUT BYTE IN OUTPUT BUFFER
    04 19 01D2 170 BRB 30$
    67 10 01D4 171 40$: BLSS 50$ ;IF LSS END OF LINE
    F1 11 01D6 172 BSBB CONVERT_R6 ;CONVERT NEXT REGISTER VALUE
    1F 10 01D8 173 BRB 30$
    EC 54 F5 01DA 174 50$: BSBB PUT ;OUTPUT RECORD
    1A 10 01DD 175 SOBGTR R4,30$ ;ANY MORE LINES TO PRINT?
    04 01DF 176 BSBB PUT ;OUTPUT RECORD
    RET
```

```
01E0 179 .SUBTITLE UTILITY SUBROUTINES
01E0 180 :
01E0 181 : INSERT TEXT IN OUTPUT BUFFER
01E0 182 :
01E0 183 :
01E0 184 .ENABL LSB
50 81 9A 01E0 185 INSERT: MOVZBL (R1)+,R0 ;GET NEXT BYTE FROM TEXT STRING
OF 13 01E3 186 BEQL 10$ ;IF EQL END OF TEXT STRING
02 10 01E5 187 BSBB PUTB ;PUT BYTE IN OUTPUT BUFFER
F7 11 01E7 188 BRB INSERT ;
01E9 189 :
01E9 190 :
01E9 191 : PUT BYTE IN OUTPUT BUFFER
01E9 192 :
01E9 193 :
22 AA B5 01E9 194 PUTB: TSTW RAB$W_RSZ(R10) ;ANY ROOM LEFT IN BUFFER?
06 13 01EC 195 BEQL 10$ ;IF EQL NO
22 AA B7 01EE 196 DECB RAB$W_RSZ(R10) ;DECREMENT NUMBER OF BYTES REMAINING
87 50 90 01F1 197 MOVB R0,(R7)+ ;INSERT BYTE IN OUTPUT BUFFER
05 01F4 198 10$: RSB ;
01F5 199 .DSABL LSB ;
01F5 200 :
01F5 201 :
01F5 202 : INSERT TEXT IN OUTPUT BUFFER AND OUTPUT RECORD AND BLANK RECORD
01F5 203 :
01F5 204 :
E9 10 01F5 205 PUTI: BSBB INSERT ;INSERT TEXT IN BUFFER
00 10 01F7 206 BSBB PUT ;OUTPUT RECORD
01F9 207 :
01F9 208 :
01F9 209 : OUTPUT CURRENT RECORD AND RESET RECORD PARAMETERS
01F9 210 :
01F9 211 :
50 57 28 AA C3 01F9 212 PUT: SUBL3 RAB$L_RBF(R10),R7,R0 ;CALCULATE LENGTH OF OUTPUT RECORD
22 AA 50 B0 01FE 213 MOVW R0,RAB$W_RSZ(R10) ;SET LENGTH OF OUTPUT RECORD
22 AB 50 B0 0202 214 MOVW R0,RAB$W_RSZ(R11) ;SET LENGTH OF OUTPUT RECORD
0206 215 $PUT RAB=(R10) ;OUTPUT RECORD
57 28 AA D0 020F 216 MOVL RAB$L_RBF(R10),R7 ;RESET OUTPUT RECORD POINTER
22 AA 50 8F 9B 0213 217 MOVZBW #BUFSIZ,RAB$W_RSZ(R10) ;RESET OUTPUT RECORD BUFFER SIZE
02 AB 02 AA B1 0218 218 CMPW RAB$W_ISI(R10),RAB$W_ISI(R11) ;ERROR AND OUTPUT FILES EQUIVALENT?
09 13 021D 219 BEQL 10$ ;IF EQL YES
05 0228 220 $PUT RAB=(R11) ;OUTPUT RECORD
0229 221 10$: RSB ;
0229 222 :
0229 223 :
0229 224 : CONVERT HEXDECIMAL VALUE AND OUTPUT RECORD
0229 225 :
0229 226 :
0229 227 PUTC_R6:
12 10 0229 228 BSBB CONVERT_R6 ;CONVERT HEXADECIMAL VALUE
CC 11 022B 229 BRB PUT ;OUTPUT RECORD
022D 230 :
022D 231 PUTC_BOTH:
B1 10 022D 232 BSBB INSERT ;INSERT INDENT PREFIX IN OUTPUT BUFFER
000B 30 022F 233 BSBW CONVERT_R6 ;CONVERT NUMBER AND PUT INTO RECORD
51 FE82 CF 9E 0232 234 MOVAB SIGCTL,R1 ;ADD HORIZONTAL SPACER
A7 10 0237 235 BSBB INSERT ;ADD SPACER TO OUTPUT RECORD
```



```
07 10 0239 236 PUTC_R8:
BC 11 0239 237 BSBB CONVERT_R8 ;CONVERT HEXDECIMAL VALUE
023B 238 BRB PUT ;OUTPUT RECORD
023D 239
023D 240 ::
023D 241 :: CONVERT NUMBER BASE 16
023D 242 ::
023D 243
023D 244 .ENABLE LOCAL_BLOCK
023D 245
023D 246 CONVERT_R6: ;LIST TRAVERSED BY R6
51 86 D0 023D 247 MOVL (R6)+,R1 ;GET NUMBER TO BE CONVERTED
03 11 0240 248 BRB 5$ ;JOIN COMMON CODE
0242 249
0242 250 CONVERT_R8: ;LIST TRAVERSED BY R8
51 88 D0 0242 251 MOVL (R8)+,R1 ;GET NUMBER TO BE CONVERTED
52 1C D0 0245 252 5$: MOVL #28,R2 ;SET STARTING BIT NUMBER
50 51 04 52 EF 0248 253 10$: EXTZV R2,#4,R1,R0 ;GET NEXT HEXDECIMAL DIGIT
50 00000000'EF40 9A 024D 254 MOVZBL EX$AB_HEX$TAB[R0],R0 ;CONVERT HEXDIGIT TO ASCII
FFEA 52 FC 8F 00 10 0255 255 BSBB PUTB ;PUT BYTE IN OUTPUT BUFFER
9D 0257 256 ACBB #0,#-4,R2,10$ ;ANY MORE DIGITS TO CONVERT?
05 025E 257 RSB ;
025F 258
025F 259 .DISABLE LOCAL_BLOCK
025F 260
025F 261 .END
```


EXCEPTMSG
Symbol table

K 10
- EXCEPTION MESSAGE FORMAT AND PRINT ROU 16-SEP-1984 00:07:22 VAX/VMS Macro V04-00
5-SEP-1984 03:41:50 [SYS.SRC]EXCEPTMSG.MAR;1

Page 7
(1)

\$\$TMP1	= 00000001		
\$\$TMP2	= 0000006B		
BUFSIZ	= 00000050		
CHFSL_MCHARGLST	= 00000008		
CHFSL_MCH_SAVRO	= 0000000C		
CHFSL_SIGARGLST	= 00000004		
CONVERT_R6	0000023D	R	02
CONVERT_R8	00000242	R	02
EXESAB_REXTAB	*****	X	02
EXESEXMSG	00000100	RG	02
EXESOPEN_MSG	*****	X	02
HEADNG	00000000	R	02
INSERT	000001E0	R	02
MESSAGE	= 00000004		
PUT	000001F9	R	02
PUTB	000001E9	R	02
PUTC_BOTH	0000022D	R	02
PUTC_R6	00000229	R	02
PUTC_R8	00000239	R	02
PUTI	000001F5	R	02
RABSL_RBF	= 00000028		
RABSW_ISI	= 00000002		
RABSW_RSZ	= 00000022		
REGDMP	00000030	R	02
REGMSG	00000021	R	02
REG_DUMP	000001B5	R	02
SIGCTL	000000B8	R	02
SIGCTL_BY_3	000000BC	R	02
SIGMSG	000000C3	R	02
SIGNAL	= 00000008		
SIGNAM	000000EA	R	02
SIGNUM	000000F5	R	02
SYSSPUT	*****	GX	02

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes														
. ABS .	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
\$ABSS	00000000 (0.)	01 (1.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE				
YEXEPAGED2	0000025F (607.)	02 (2.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE				

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.09	00:00:01.00
Command processing	105	00:00:00.49	00:00:04.34
Pass 1	181	00:00:03.75	00:00:11.25
Symbol table sort	0	00:00:00.32	00:00:00.84
Pass 2	71	00:00:00.96	00:00:07.92
Symbol table output	6	00:00:00.06	00:00:00.06
Psect synopsis output	1	00:00:00.02	00:00:00.02

EXCEPTMSG
VAX-11 Macro Run Statistics

- EXCEPTION MESSAGE FORMAT AND PRINT ROU L 10
16-SEP-1984 00:07:22 VAX/VMS Macro V04-00
5-SEP-1984 03:41:50 [SYS.SRC]EXCEPTMSG.MAR;1

Page 8
(1)

Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	396	00:00:05.70	00:00:25.47

The working set limit was 1050 pages.
21155 bytes (42 pages) of virtual memory were used to buffer the intermediate code.
There were 20 pages of symbol table space allocated to hold 339 non-local and 12 local symbols.
261 source lines were read in Pass 1, producing 16 object records in Pass 2.
13 pages of virtual memory were used to define 11 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
-----	-----
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
-\$255\$DUA28:[SYS.LIB]STARLET.MLB;2	8
TOTALS (all libraries)	8

427 GETS were required to define 8 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:EXCEPTMSG/OBJ=OBJ\$:EXCEPTMSG MSRC\$:EXCEPTMSG/UPDATE=(ENH\$:EXCEPTMSG)+EXECMLS/LIB

0374 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

