


```

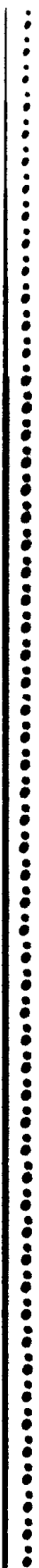
MM      MM      AAAAAA  RRRRRRRR  88888888  LL      IIIIII
MM      MM      AAAAAA  RRRRRRRR  88888888  LL      IIIIII
MMMM    MMMM    AA      AA  RR      RR  88      88  LL      II
MMMM    MMMM    AA      AA  RR      RR  88      88  LL      II
MM      MM      AA      AA  RR      RR  88      88  LL      II
MM      MM      A      A   RR      RR  88      88  LL      II
MM      MM      AA      AA  RRRRRRRR  88888888  LL      II
MM      MM      AA      AA  RRRRRRRR  88888888  LL      II
MM      MM      AAAAAAAAAA RR      RR  88      88  LL      II
MM      MM      AAAAAAAAAA RR      RR  88      88  LL      II
MM      MM      AA      AA  RR      RR  88      88  LL      II
MM      MM      AA      AA  RR      RR  88      88  LL      II
MM      MM      AA      AA  RR      RR  88888888  LLLLLLLLLL  IIIIII
MM      MM      AA      AA  RR      RR  88888888  LLLLLLLLLL  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

```



```

1 0001 0 MODULE MARBLI(IDENT='V04-000',MAIN=CONVERT %TITLE'MARS to BLISS Macro Converter')=
2 0002 1 BEGIN
3 0003 1
4 0004 1 *****
5 0005 1 *
6 0006 1 *   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
7 0007 1 *   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
8 0008 1 *   ALL RIGHTS RESERVED.
9 0009 1 *
10 0010 1 *   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
11 0011 1 *   ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
12 0012 1 *   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
13 0013 1 *   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
14 0014 1 *   OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
15 0015 1 *   TRANSFERRED.
16 0016 1 *
17 0017 1 *   THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
18 0018 1 *   AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
19 0019 1 *   CORPORATION.
20 0020 1 *
21 0021 1 *   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
22 0022 1 *   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
23 0023 1 *
24 0024 1 *
25 0025 1 *****
26 0026 1
27 0027 1 ++
28 0028 1
29 0029 1 ABSTRACT:
30 0030 1   The routines in this module, along with XPORT
31 0031 1   convert a specifically formatted MARS macro file into a
32 0032 1   BLISS macro file.
33 0033 1
34 0034 1   The logic in this module is somewhat ad hoc and
35 0035 1   any perturbation of the format of the MARS file may result
36 0036 1   in these routines not working.
37 0037 1
38 0038 1 AUTHOR:
39 0039 1   P.C. Marks, CREATION DATE: 3 FEB 77
40 0040 1
41 0041 1 MODIFIED BY:
42 0042 1
43 0043 1   V03-001 MLJ0093      Martin L. Jack, 14-Jul-1982 11:38
44 0044 1   If the ".MACRO" line contains the string "CJFS", change the
45 0045 1   routine prefix from SYSS to CJFS. Do the same for RUF.
46 0046 1
47 0047 1   V02-003      BLS0073      Benn Schreiber      26-Aug-1981
48 0048 1   Disable per-macro output
49 0049 1
50 0050 1   V02-002      APL0001      Al Lehotsky      4-Jun-1981
51 0051 1   Change for 31-character names which was missed earlier.
52 0052 1
53 0053 1   V02-001      BLS0055      Benn Schreiber      3-Jun-1981
54 0054 1   Use BLISS linkage and GENERAL addressing mode
55 0055 1
56 0056 1   A.P. Lehotsky 31-Oct-79      Transport to VAX. Suppress "()" in
57 0057 1   macros without arguments, e.g. $HIBER

```

```

: 58      0058 1  |
: 59      0059 1  |--
: 60      0060 1
: 61      0061 1
: 62      0062 1 FORWARD ROUTINE
: 63      0063 1   CONVERT,
: 64      0064 1   CONVMACRO:      NOVALUE,
: 65      0065 1   OUTLIST,
: 66      0066 1   CONVARGLIST:   NOVALUE;
: 67      0067 1
: 68      0068 1 MACRO

```

```

: Main routine
: Convert a .MACRO definition
: Output a parameter list
: Convert argument list of a .MACRO

```

```

69 0069 1 VERSTR= 'MARBLI V03-001' %;
70 0070 1 LIBRARY 'SYSS$LIBRARY:XPORT';
71 0071 1
72 0072 1 LITERAL
73 0073 1 MAX_NO_ARGS= 63; ! Max number of macro arguments
74 0074 1 MAX_ARG_LIST= 65*MAX_NO_ARGS; ! Max characters in formal arg list
75 0075 1 MAX_REC_SIZE= 130; ! Max input and output record length
76 0076 1
77 0077 1 MACRO
78 M 0078 1 REPEAT=
79 0079 1 WHILE 1 DO %,
80 0080 1
81 0081 1 ! Construct length and pointer parameters
82 0082 1
83 M 0083 1 CH$LEN PTR[]=
84 0084 1 %CH$RCOUNT(%REMAINING), CH$PTR(UPLIT(%REMAINING)) %,
85 0085 1
86 0086 1
87 0087 1 ! Return the index of string S in context C
88 0088 1
89 M 0089 1 CH$INDEX(CL,C,S) =
90 0090 1 CH$FIND_SUB(CL, C, %CH$RCOUNT(S), CH$PTR(UPLIT(S))) %,
91 0091 1
92 0092 1 ! Return the size of the zero-truncated output buffer
93 0093 1
94 M 0094 1 TRUNCATED_OUTPUT =
95 0095 1 CH$DIFF(CH$FIND_CH(MAX_REC_SIZE+1, CH$PTR(OUTPUT_RECORD),0),
96 0096 1 CH$PTR(OUTPUT_RECORD)) %;
97 0097 1
98 0098 1
99 0099 1 OWN
100 0100 1 !+
101 0101 1 ! Declarations for processing a text line for macro declaration and
102 0102 1 ! the associated argument list.
103 0103 1 !-
104 0104 1 INPUT_RECORD: VECTOR[CH$ALLOCATION(MAX_REC_SIZE)],
105 0105 1 INPUT_PTR,
106 0106 1 INPUT_LENGTH,
107 0107 1
108 0108 1 ! Guarantee at least 1 zero byte at end of output
109 0109 1 ! record s.t. TRUNCATED_OUTPUT macro cannot fail.
110 0110 1
111 0111 1 OUTPUT_RECORD: VECTOR[CH$ALLOCATION(MAX_REC_SIZE+1)],
112 0112 1 OUTPUT_PTR,
113 0113 1
114 0114 1 ARG_PTR,
115 0115 1 ARG_LENGTH,
116 0116 1 ARG_LIST: VECTOR[CH$ALLOCATION(MAX_ARG_LIST)],
117 0117 1
118 0118 1 CALL_PTR,
119 0119 1 CALL_LENGTH,
120 0120 1 CALL_LIST: VECTOR[CH$ALLOCATION(MAX_ARG_LIST)],
121 0121 1
122 0122 1 VAR_ARGS;
123 0123 1
124 0124 1 OWN !+
125 0125 1

```

```
: 126      0126 1      ; I/O related declarations.  
: 127      0127 1      ;  
: 128      0128 1      ;  
: 129      0129 1      TERMINAL:  $XPO_IOB(),  
: 130      0130 1      INPUT:    $XPO_IOB(),  
: 131      0131 1      OUTPUT:   $XPO_IOB();
```

```

133 0132 1 ROUTINE CONVERT =
134 0133 1
135 0134 1 **
136 0135 1 FUNCTIONAL DESCRIPTION:
137 0136 1 This is the main routine of this module.
138 0137 1 The chief function is to examine an input line and determine
139 0138 1 whether it should be ignored or processed by other routines.
140 0139 1 IMPLICIT INPUTS:
141 0140 1
142 0141 1 OWN storage
143 0142 1
144 0143 1 IMPLICIT OUTPUTS:
145 0144 1
146 0145 1 OWN storage
147 0146 1
148 0147 1 ROUTINE VALUE:
149 0148 1
150 0149 1 Success or an XPORT completion code
151 0150 1 --
152 0151 2 BEGIN
153 0152 2 LOCAL
154 0153 2 PRINT_COMMENTS; ! Flag to pass comments and blank lines to output
155 0154 2
156 0155 2 $XPO_IOB_INIT(IOB=TERMINAL);
157 0156 2 $XPO_IOB_INIT(IOB=INPUT);
158 0157 2 $XPO_IOB_INIT(IOB=OUTPUT);
159 0158 2
160 0159 2 $XPO_OPEN( IOB=TERMINAL, FILE_SPEC=$XPO_INPUT);
161 0160 2 $XPO_PUT( IOB=TERMINAL,
162 0161 2 !
163 0162 2 STRING=(%STRING('System-Service Macro Translator ',%EXPAND VERSTR)) );
164 0163 2 REPEAT
165 0164 3 BEGIN
166 0165 4 IF NOT $XPO_GET(IOB=TERMINAL, PROMPT=('Input file (STARLET.MAR): '))
167 0166 3 THEN
168 0167 3 RETURN XPOS_NORMAL;
169 0168 3
170 P 0169 3 IF $XPO_OPEN(IOB=INPUT,
171 P 0170 3 FILE_SPEC=TERMINAL[IOBST_STRING], ! User supplied name
172 P 0171 3 DEFAULT=('STARLET.MAR'), ! The standard input
173 0172 4 FAILURE=XPOSIO_FAILURE)
174 0173 3 THEN
175 0174 3 EXITLOOP ! Got good input.
176 0175 2 END;
177 0176 2
178 0177 3 IF NOT $XPO_GET(IOB=TERMINAL, PROMPT=('Output file (*.B32): '))
179 0178 2 THEN
180 0179 2 RETURN XPOS_NORMAL;
181 0180 2
182 P 0181 2 $XPO_OPEN(IOB=OUTPUT,
183 P 0182 2 FILE_SPEC=TERMINAL[IOBST_STRING],
184 P 0183 2 DEFAULT=('*.B32'),
185 P 0184 2 RELATED=INPUT[IOBST_RESULTANT],
186 0185 2 OPTION=OUTPUT);
187 0186 2
188 P 0187 2 $XPO_PUT(IOB=OUTPUT,

```



```

: 246      0245 2
: 247      0246 2 $XPO_CLOSE(IOB=INPUT);
: 248      0247 2 $XPO_CLOSE(IOB=OUTPUT);
: 249      0248 2
: 250      0249 2 ! $XPO_PUT( IOB=TERMINAL, STRING=('End MARBLI') );
: 251      0250 2 $XPO_CLOSE( IOB=TERMINAL );
: 252      0251 2
: 253      0252 2 XPOS_NORMAL
: 254      0253 1 END;

```

```

                                .TITLE MARBLI MARS to BLISS Macro Converter
                                .IDENT  \V04-000\
                                .PSECT  $SPLITS,NOVRT,NOEXE,2
41 54 53 28 20 65 6C 69 66 20 74 75 70 6E 49 00000 P.AAA: .ASCII  \SYSS$INPUT\
: 20 3A 29 52 41 4D 2E 54 45 4C 52 00009 P.AAB: .ASCII  \Input file (STARLET.MAR): \
: 52 41 4D 2E 54 45 4C 52 41 54 53 00018
2E 2A 28 20 65 6C 69 66 20 74 75 70 74 75 4F 00023 P.AAC: .ASCII  \STARLET.MAR\
: 20 3A 29 32 33 42 0002E P.AAD: .ASCII  \Output file (*.B32): \
: 32 33 42 2E 2A 00030
72 66 20 64 65 74 61 6C 73 6E 61 72 54 20 21 00043 P.AAE: .ASCII  \*.B32\
: 20 79 62 20 32 33 2D 4F 52 43 41 4D 20 6D 6F 00048 P.AAF: .ASCII  \! Translated from MACRO-32 by MARBLI V03\
: 33 30 56 20 49 4C 42 52 41 4D 00057
: 31 30 30 2D 00066
: 00 00 4F 52 43 41 4D 2E 00070 .ASCII  \-001\
: 00 00 4F 52 43 41 4D 2E 00074 P.AAG: .BLKB  0
: 00 00 4F 52 43 41 4D 2E 00074 P.AAH: .ASCII  \.MACRO\<0><0>
: 00 00 4F 52 43 41 4D 2E 0007C P.AAI: .ASCII  \_S\<0><0>
                                .PSECT  $OWNS,NOEXE,2
                                00000 INPUT_RECORD:
                                .BLKB  132
                                00084 INPUT_PTR:
                                .BLKB   4
                                00088 INPUT_LENGTH:
                                .BLKB   4
                                0008C OUTPUT_RECORD:
                                .BLKB  132
                                00110 OUTPUT_PTR:
                                .BLKB   4
                                00114 ARG_PTR: .BLKB   4
                                00118 ARG_LENGTH:
                                .BLKB   4
                                0011C ARG_LIST:
                                .BLKB  4096
                                0111C CALL_PTR:
                                .BLKB   4
                                01120 CALL_LENGTH:
                                .BLKB   4
                                01124 CALL_LIST:
                                .BLKB  4096
                                02124 VAR_ARGS:
                                .BLKB   4
                                02128 TERMINAL:

```

```

                                .BLKB 244
0221C INPUT: .BLKB 244
02310 OUTPUT: .BLKB 244
0009 02404 $IOB$FILE_SPEC:
                                .WORD 9
01 0E 02406 .BYTE 14, 1
00000000' 02408 .ADDRESS P.AAA
001A 0240C $IOB$PROMPT:
                                .WORD 26
01 0E 0240E .BYTE 14, 1
00000000' 02410 .ADDRESS P.AAB
000B 02414 $IOB$DEFAULT:
                                .WORD 11
01 0E 02416 .BYTE 14, 1
00000000' 02418 .ADDRESS P.AAC
0015 0241C $IOB$PROMPT:
                                .WORD 21
01 0E 0241E .BYTE 14, 1
00000000' 02420 .ADDRESS P.AAD
0005 02424 $IOB$DEFAULT:
                                .WORD 5
01 0E 02426 .BYTE 14, 1
00000000' 02428 .ADDRESS P.AAE
002C 0242C $IOB$OUTPUT:
                                .WORD 44
01 0E 0242E .BYTE 14, 1
00000000' 02430 .ADDRESS P.AAF
0000 02434 $IOB$OUTPUT:
                                .WORD 0
01 0E 02436 .BYTE 14, 1
00000C00' 02438 .ADDRESS P.AAG

```

```

IOB$= TERMINAL
IOB$RESULTANT= TERMINAL+28
IOB$= INPUT
IOB$RESULTANT= INPUT+28
IOB$= OUTPUT
IOB$RESULTANT= OUTPUT+28
IOB$= TERMINAL
IOB$= TERMINAL
IOB$= INPUT
$IOB$FILE_SPEC= TERMINAL+52
IOB$= TERMINAL
IOB$= OUTPUT
$IOB$FILE_SPEC= TERMINAL+52
$IOB$RELATED= INPUT+28
IOB$= OUTPUT
IOB$= INPUT
IOB$= OUTPUT
IOB$= OUTPUT
IOB$= INPUT
IOB$= OUTPUT
IOB$= TERMINAL
.EXTRN XPOS$OPEN, XPOS$FAILURE
.EXTRN XST$FREE TEMP, XPOS$GET
.EXTRN XPOS$IO FAILURE, XPOS$PUT
.EXTRN XPOS$CLOSE

```

.....

.....

.PSECT \$CODE\$,NOWRT,2

				OFFC 00000	CONVERT: .WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	0132
			5B 00000000G	EF 9E 00002	MOVAB	XPOSGET, R11	
			5A 00000000G	EF 9E 00009	MOVAB	XPOSOPEN, R10	
			59 00000000G	EF 9E 00010	MOVAB	XPOSFAILURE, R9	
			58 0000'	CF 9E 00017	MOVAB	IOBS, R8	
00F4	8F	00	5E	08 C2 0001C	SUBL2	#8, SP	
			6E	00 2C 0001F	MOVCS	#0, (SP), #0, #244, IOBS	0155
				C8 00026			
		FE18	C8 0301003D	8F D0 00029	MOVL	#50397245, IOBS	
00F4	8F	00	FE36	C8 020E	MOVW	#526, IOBSRESULTANT+2	
			6E	00 2C 00039	MOVCS	#0, (SP), #0, #244, IOBS	0156
				C8 00040			
		FF0C	C8 0301003D	8F D0 C0043	MOVL	#50397245, IOBS	
00F4	8F	00	FF2A	C8 020E	MOVW	#526, IOBSRESULTANT+2	
			6E	00 2C 00053	MOVCS	#0, (SP), #0, #244, IOBS	0157
				68 0005A			
		1E	68 0301003D	8F D0 0005B	MOVL	#50397245, IOBS	
		FE1C	A8 020E	8F B0 00062	MOVW	#526, IOBSRESULTANT+2	
		FE44	C8 00F4	C8 9E 00068	MOVAB	\$IOBSFILE_SPEC, IOBS+4	0159
				01 90 0006F	MOVB	#1, IOBS+44	
				59 DD 00074	PUSHL	R9	
				7E D4 00076	CLRL	-(SP)	
		FE18	C8 9F 00078	C8 9F 00078	PUSHAB	IOBS	
		6A	03 FB 0007C	03 FB 0007C	CALLS	#3, XPOSOPEN	
		50	C8 D0 0007F	C8 D0 0007F	MOVL	IOBS+36, R0	0165
			09 13 00084	09 13 00084	BEQL	2\$	
			50 DD 00086	50 DD 00086	PUSHL	R0	
		00000000G	EF FB 00088	01 FB 00088	CALLS	#1, XST\$FREE_TEMP	
		FE3C	C8 00FC	C8 9E 0008F	MOVAB	\$IOBSPROMPT, -IOBS+36	
		FE44	C8	06 90 00096	MOVB	#6, IOBS+44	
				59 DD 0009B	PUSHL	R9	
				7E D4 0009D	CLRL	-(SP)	
		FE18	C8 9F 0009F	C8 9F 0009F	PUSHAB	IOBS	
		6B	03 FB 000A3	03 FB 000A3	CALLS	#3, XPOSGET	
		4C	50 E9 000A6	50 E9 000A6	BLBC	R0, 4\$	
		FF10	C8 FE4C	C8 9E 000A9	MOVAB	\$IOBSFILE_SPEC, IOBS+4	0172
		FF14	C8 0104	C8 9E 000B0	MOVAB	\$IOBSDEFAULT, IOBS+8	
		FF38	C8	01 90 000B7	MOVB	#1, IOBS+44	
			00000000G	EF 9F 000BC	PUSHAB	XPOSIO_FAILURE	
				7E D4 000C2	CLRL	-(SP)	
		FF0C	C8 9F 000C4	C8 9F 000C4	PUSHAB	IOBS	
		6A	03 FB 000C8	03 FB 000C8	CALLS	#3, XPOSOPEN	
		B1	50 E9 000CB	50 E9 000CB	BLBC	R0, 1\$	
		50	C8 D0 000CE	C8 D0 000CE	MOVL	IOBS+36, R0	0177
			09 13 000D3	09 13 000D3	BEQL	3\$	
			50 DD 000D5	50 DD 000D5	PUSHL	R0	
		00000000G	EF FB 000D7	01 FB 000D7	CALLS	#1, XST\$FREE_TEMP	
		FE3C	C8 010C	C8 9E 000DE	MOVAB	\$IOBSPROMPT, -IOBS+36	
		FE44	C8	06 90 000E5	MOVB	#6, IOBS+44	
				59 DD 000EA	PUSHL	R9	
				7E D4 000EC	CLRL	-(SP)	
		FE18	C8 9F 000EE	C8 9F 000EE	PUSHAB	IOBS	
		6B	03 FB 000F2	03 FB 000F2	CALLS	#3, XPOSGET	
		03	50 E8 000F5	50 E8 000F5	BLBS	R0, 5\$	

				7E	D4	001D3		CLRL	-(SP)		
				58	DD	001D5		PUSHL	R8		
		00000000G	EF	03	FB	001D7		CALLS	#3, XPOS\$PUT		
			50					MOVL	OUTPUT_PTR, R0		0229
			2A	DE00	C8	D0	001DE	CMPB	1(R0), #42		
				01	A0	91	001E3	BNEQ	9\$		
			57		01	D0	001E9	MOVL	#1, PRINT_COMMENTS		0231
					97	11	001EC	BRB	9\$		0229
66	DD78	C8	0000'	CF	06	39	001EE	MATCHC	#6, P.AAH, INPUT_LENGTH, (R6)		0234
					03	13	001F7	BEQL	14\$		
			53		06	D0	001F9	MOVL	#6, R3		
			53		06	C2	001FC	SUBL2	#6, R3		
					84	13	001FF	BEQL	9\$		
66	DD78	C8	0000'	CF	02	39	00201	MATCHC	#2, P.AAI, INPUT_LENGTH, (R6)		0236
					03	13	0020A	BEQL	15\$		
			53		02	D0	0020C	MOVL	#2, R3		
			53		02	C2	0020F	SUBL2	#2, R3		
					05	13	00212	BEQL	16\$		
			0000V	CF	00	FB	00214	CALLS	#0, CONVMACRO		0241
					FF22	31	00219	BRW	6\$		0206
			FF38	C8	02	90	0021C	MOVB	#2, IOB\$+44		0246
					59	DD	00221	PUSHL	R9		
					7E	D4	00223	CLRL	-(SP)		
					C8	9F	00225	PUSHAB	IOB\$		
		00000000G	EF	FF0C	03	FB	00229	CALLS	#3, XPOS\$CLOSE		
			2C	A8	02	90	00230	MOVB	#2, IOB\$+44		0247
					59	DD	00234	PUSHL	R9		
					7E	D4	00236	CLRL	-(SP)		
					58	DD	00238	PUSHL	R8		
		00000000G	EF		03	FB	0023A	CALLS	#3, XPOS\$CLOSE		
			FE44	C8	02	90	00241	MOVB	#2, IOB\$+44		0250
					59	DD	00246	PUSHL	R9		
					7E	D4	00248	CLRL	-(SP)		
					C8	9F	0024A	PUSHAB	IOB\$		
		00000000G	EF	FE18	03	FB	0024E	CALLS	#3, XPOS\$CLOSE		
			50	00208001	8F	D0	00255	MOVL	#2129921, R0		0253
					04	0025C		RET			

; Routine Size: 605 bytes, Routine Base: \$CODE\$ + 0000

```
256 0254 1 ROUTINE CONVMACRO: NOVALUE=  
257 0255 1  
258 0256 1 !++  
259 0257 1 FUNCTIONAL DESCRIPTION:  
260 0258 1 This routine reads and processes a MARS system macro  
261 0259 1 It calls the routine CONVARGLIST to convert the argument list  
262 0260 1 of the macro definition.  
263 0261 1  
264 0262 1 PARAMETERS:  
265 0263 1  
266 0264 1 NONE  
267 0265 1  
268 0266 1 ROUTINE VALUE:  
269 0267 1  
270 0268 1 NONE  
271 0269 1  
272 0270 1 !--  
273 0271 1  
274 0272 2 BEGIN  
275 0273 2 LOCAL  
276 0274 2 PREFIX_PTR,  
277 0275 2 PTR, ! Temp CH$ pointer for copying arglists  
278 0276 2 TEMP_PTR1,  
279 0277 2 TEMP_PTR2,  
280 0278 2 TEMPSTRING: VECTOR[CH$ALLOCATION(80)],  
281 0279 2 MACRO_PTR,  
282 0280 2 MACRO_LENGTH,  
283 0281 2 MACRO_NAME: VECTOR[CH$ALLOCATION(31)];  
284 0282 2  
285 0283 2  
286 0284 2 MACRO_PTR = CH$PTR(MACRO_NAME);  
287 0285 2  
288 0286 2  
289 0287 2 ! If the .MACRO line contains the string "CJFS", then change the routine  
290 0288 2 ! name prefix from SY$$ to CJFS. Do the same for RUF.  
291 0289 2  
292 0290 2 PREFIX_PTR = CH$PTR(UPLIT BYTE('SYS'));  
293 0291 2 IF CH$FIND SUB(MAX_REC_SIZE, .OUTPUT_PTR, CH$LEN_PTR('CJFS')) NEQ 0  
294 0292 2 THEN PREFIX_PTR = CH$PTR(UPLIT BYTE('CJF'));  
295 0293 2 IF CH$FIND SUB(MAX_REC_SIZE, .OUTPUT_PTR, CH$LEN_PTR('RUF$')) NEQ 0  
296 0294 2 THEN PREFIX_PTR = CH$PTR(UPLIT BYTE('RUF'));  
297 0295 2  
298 0296 2  
299 0297 2 ! Search for the '$' which begins the macro name and  
300 0298 2 ! the '_' which ends it, and initialize MACRO_NAME and MACRO_LENGTH  
301 0299 2  
302 0300 2 TEMP_PTR1 = CH$FIND_CH(MAX_REC_SIZE, .OUTPUT_PTR, %C'$');  
303 0301 2 TEMP_PTR2 = CH$FIND_CH(MAX_REC_SIZE, .OUTPUT_PTR, %C'_');  
304 0302 2 MACRO_LENGTH = CH$DIFF(.TEMP_PTR2, .TEMP_PTR1);  
305 0303 2 CH$MOVE(.MACRO_LENGTH, .TEMP_PTR1, .MACRO_PTR);  
306 0304 2  
307 0305 2  
308 0306 2 ! Inform user of progress  
309 0307 2  
310 0308 2 ! CH$COPY(CH$LEN_PTR('Macro: '),  
311 0309 2 ! .MACRO_LENGTH, .MACRO_PTR,  
312 0310 2 ! 0,  
312 0311 2 !
```

```
313      80, CH$PTR(TEMPSTRING) );
314
315      $XPO_PUT( IOB=TERMINAL, STRING=(80, CH$PTR(TEMPSTRING) ) );
316
317
318      ! Gather the argument list
319      !
320      CONVARGLIST(.TEMP_PTR2);
321
322
323      ! Put out one of the lines
324      "KEYWORDMACRO $name(formal=default,...)="
325      or
326      "MACRO $name[]" if VAR_ARGS is true
327
328      IF .VAR_ARGS
329      THEN
330          CH$COPY(
331              CH$LEN_PTR('MACRO '),
332              .MACRO_LENGTH,
333              .MACRO_PTR,
334              CH$LEN_PTR('[]='),
335              0,
336              MAY_REC_SIZE,
337              .OUTPUT_PTR)
338      ELSE
339          BEGIN
340          LOCAL
341          PTR;
342
343          PTR = CH$MOVE(CH$LEN_PTR('KEYWORDMACRO '), .OUTPUT_PTR);
344          PTR = CH$MOVE(.MACRO_LENGTH, .MACRO_PTR, .PTR);
345
346          IF .ARG_LENGTH GTR 0          ! If there are formal parameters, then
347          THEN                          ! put out the argument list.
348              BEGIN
349              PTR = CH$MOVE( CH$LEN_PTR('('), .PTR );
350              PTR = OUTLIST( .PTR, .ARG_LENGTH, .ARG_PTR);
351              PTR = CH$MOVE( CH$LEN_PTR(')='), .PTR)
352              END
353          ELSE
354              CH$WCHAR_A(%C'=', PTR);      ! There were no formal parameters
355
356          CH$WCHAR(0, .PTR)                ! Mark end of string
357          END;
358
359
360      $XPO_PUT(IOB=OUTPUT,
361              STRING=(TRUNCATED_OUTPUT, .OUTPUT_PTR) );
362
363
364      ! Put out the line " ("
365      !
366      $XPO_PUT(IOB=OUTPUT, STRING=(' ('));
367
368
369      ! Put out the line " EXTERNAL ROUTINE SYS$name: BLISS ADDRESSING_MODE(GENERAL);"
```

```

: 370 0368 2 !
: 371 0369 2 CH$COPY(
: 372 0370 2 CH$LEN_PTR(' EXTERNAL ROUTINE '),
: 373 0371 2 3, .PREFIX_PTR,
: 374 0372 2 .MACRO_LENGTH,
: 375 0373 2 .MACRO_PTR,
: 376 0374 2 CH$LEN_PTR(': BLISS ADDRESSING_MODE(GENERAL);'),
: 377 0375 2 0,
: 378 0376 2 MAX_REC_SIZE,
: 379 0377 2 .OUTPUT_PTR);
: 380 0378 2 $XPO_PUT(IOB=OUTPUT, STRING=(TRUNCATED_OUTPUT, .OUTPUT_PTR) );
: 381 0379 2
: 382 0380 2 ! Put out the line " SYS$name(formal,...)"
: 383 0381 2 !
: 384 0382 2 PTR = CH$COPY(
: 385 0383 2 CH$LEN_PTR(' '),
: 386 0384 2 3, .PREFIX_PTR,
: 387 0385 2 .MACRO_LENGTH,
: 388 0386 2 .MACRO_PTR,
: 389 0387 2 CH$LEN_PTR('('),
: 390 0388 2 0,
: 391 0389 2 .MACRO_LENGTH + 8, ! Exact copying
: 392 0390 2 .OUTPUT_PTR);
: 393 0391 2
: 394 0392 2 PTR = OUTLIST( .PTR, .CALL_LENGTH, .CALL_PTR);
: 395 0393 2 PTR = CH$MOVE( CH$LEN_PTR('-'), %CHAR(0)-), .PTR );
: 396 0394 2
: 397 0395 2 $XPO_PUT( IOB=OUTPUT, STRING=(TRUNCATED_OUTPUT, .OUTPUT_PTR) );
: 398 0396 2
: 399 0397 2 ! Close up the macro declaration
: 400 0398 2 !
: 401 0399 2 $XPO_PUT(IOB=OUTPUT, STRING=(' ) %;') )
: 402 0400 2
: 403 0401 2 END;
: 404 0402 2
: 405 0403 2

```

										.PSECT	\$SPLITS	NOVRT	NOEXE	2													
										53	59	53	00080	P.AAJ:	.ASCII	\SYS\											
													00083		.BLKB	1											
										24	46	4A	43	00084	P.AAK:	.ASCII	\CJF\$\										
											46	4A	43	00088	P.AAL:	.ASCII	\CJF\										
														0008B		.BLKB	1										
										24	46	55	52	0008C	P.AAM:	.ASCII	\RUF\$\										
											46	55	52	00090	P.AAN:	.ASCII	\RUF\										
														00093		.BLKB	1										
										00	00	20	4F	52	43	41	4D	00094	P.AAO:	.ASCII	\MACRO \<0><0>						
														00	3D	5D	5B	0009C	P.AAP:	.ASCII	\[]=\<0>						
00	00	20	4F	52	43	41	4D	44	52	4F	57	59	45	4B				000A0	P.AAQ:	.ASCII	\KEYWORDMACRO \<0><0><0>						
																		00	00AF								
																		00	00B0	P.AAR:	.ASCII	\(\<0><0><0>					
																		00	00B4	P.AAS:	.ASCII	\)=\<0><0>					
																		28	20	20	20	20	000B8	P.AAT:	.ASCII	\ (\	
																			000BD		.BLKB	3					


```

4F 52 20 4C 41 4E 52 45 54 58 45 20 20 20 20 000C0 P.AAU: .ASCII \ EXTERNAL ROUTINE \<0><0><0>
53 53 45 52 44 44 00 00 00 20 45 4E 49 54 55 000CF
41 52 45 4E 45 47 28 45 44 4F 4D 5F 47 4E 49 000D8 P.AAV: .ASCII \: BLISS ADDRESSING_MODE(GENERAL);\<0><0>
00 00 3B 29 4C 000E7
00 00 00 00 000F6
20 20 20 20 000FC P.AAW: .ASCII <0>
00 00 00 28 00100 P.AAX: .ASCII \(\<0><0><0>
00 00 00 29 00104 P.AAY: .ASCII \)\<0><0><0>
00 00 00 00 00108 .ASCII <0><0><0><0>
3B 25 20 29 20 20 20 20 0010C P.AAZ: .ASCII \ ) %;\

```

.PSECT \$OWNS,NOEXE,2

```

0005 0243C $IOBS$OUTPUT:
          .WORD 5
01 0E 0243E .BYTE 14, 1
00000000 02440 .ADDRESS P.AAT
0008 02444 $IOBS$OUTPUT:
          .WORD 8
01 0E 02446 .BYTE 14, 1
00000000 02443 .ADDRESS P.AAZ

```

```

IOBS= OUTPUT
IOBS= OUTPUT
IOBS= OUTPUT
IOBS= OUTPUT
IOBS= OUTPUT

```

.PSECT \$CODE\$,NOWRT,2

```

OFFC 00000 CONVMACRO:
          .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11
5B 0000 CF 9E 00002 MOVAB OUTPUT_PTR, R11
5E 88 AE 9E 00C07 MOVAB -120(SP), SP
59 08 AE 9E 0000B MOVAB MACRO_NAME, MACRO_PTR
5A 0000 CF 9E 0000F MOVAB P.AAJ, PREFIX_PTR
64 0082 8F 0000 CF 6B D0 00014 MOVL OUTPUT_PTR, R4
          04 39 00017 MATCHC #4, P.AAK, #130, (R4)
          03 13 00020 BEQL 1$
53 04 D0 00022 MOVL #4, R3
53 04 C2 00025 1$: SUBL2 #4, R3
          05 13 00028 BEQL 2$
64 0082 8F 0000 SA 0000 CF 9E 0002A MOVAB P.AAL, PREFIX_PTR
          04 39 0002F 2$: MATCHC #4, P.AAM, #130, (R4)
          03 13 00038 BEQL 3$
53 04 D0 0003A MOVL #4, R3
53 04 C2 0003D 3$: SUBL2 #4, R3
          05 13 00040 BEQL 4$
          SA 0000 CF 9E 00042 MOVAB P.AAN, PREFIX_PTR
64 0082 8F 24 3A 00047 4$: LOCC #36, #130, (R4)
          02 12 0004D BNEQ 5$
          51 D4 0004F CLRL R1
          SA 0000 CF 9E 00051 5$: MOVL R1, TEMP_PTR1
64 0082 8F 5F 8F 3A 00054 LOCC #95, #130, (R4)
          02 12 0005B BNEQ 6$

```

				51	D4	0005D			CLRL	R1		
				51	D0	0005F	6\$:		MOVL	R1, TEMP_PTR2		
	56			52	C3	00062			SUBL3	TEMP_PTR1, TEMP_PTR2, MACRO_LENGTH		0302
	69			56	28	00066			MOVCS	MACRO_LENGTH, (TEMP_PTR1), (MACRO_PTR)		0303
				57	DD	0006A			PUSHL	TEMP_PTR2		0318
		0000V		01	FB	0006C			CALLS	#1, CONVARGLIST		
				2F	CB	00071	2014		BLBC	VAR_ARGS, 7\$		0326
				58	8F	00076	82		MOVZBL	#130, R8		0328
58				57	6B	0007A			MOVL	OUTPUT_PTR, R7		0335
	00	0000'		06	2C	0007D			MOVCS	#6, P.AAO, #0, R8, (R7)		
				67		00084						
				4F	18	00085			BGEQ	10\$		
				06	C0	00087			ADDL2	#6, R7		
58				58	06	0008A			SUBL2	#6, R8		
	00			69	56	0008D			MOVCS	MACRO_LENGTH, (MACRO_PTR), #0, R8, (R7)		
				67	2C	00092						
				41	18	00093			BGEQ	10\$		
				56	C0	00095			ADDL2	MACRO_LENGTH, R7		
58				58	56	00098			SUBL2	MACRO_LENGTH, R8		
	00	0000'		03	2C	0009B			MOVCS	#3, P.AAP, #0, R8, (R7)		
				67		000A2						
				31	11	000A3			BRB	10\$		0328
	00	0000'		0D	28	000A5	7\$:		MOVCS	#13, P.AAQ, @OUTPUT_PTR		0341
	63			56	28	000AC			MOVCS	MACRO_LENGTH, (MACRO_PTR), (PTR)		0342
				50	AB	000B0	08		MOVL	ARG_LENGTH, R0		0344
				1B	15	000B4			BLEQ	8\$		
				83	CF	000B6	0000'		MOVB	P.AAR, (PTR)+		0347
				AB	DD	000BB	04		PUSHL	ARG_PTR		0348
				50	DD	000BE			PUSHL	R0		
				53	DD	000C0			PUSHL	PTR		
		0000V		03	FB	000C2			CALLS	#3, OUTLIST		
				50	D0	000C7			MOVL	R0, PTR		
				83	CF	000CA	0000'		MOVW	P.AAS, (PTR)+		0349
				03	11	000CF			BRB	9\$		0352
				83	3D	000D1	8\$:		MCVB	#61, (PTR)+		
				63	94	000D4	9\$:		CLRB	(PTR)		0354
FF7C	CB	0083		00	3A	000D6	10\$:		LOCC	#0, #131, OUTPUT_RECORD		0359
				02	12	000DE			BNEQ	11\$		
				51	D4	000E0			CLRL	R1		
				50	9E	000E2	FF7C		MOVAB	OUTPUT_RECORD, R0		
	6E			50	A3	000E7			SUBW3	R0, R1, \$IOB\$OUTPUT		
		02		0E	90	000EB			MOVB	#14, \$IOB\$OUTPUT+2		
		03		01	90	000EF			MOVB	#1, \$IOB\$OUTPUT+3		
		04		6B	D0	000F3			MOVL	OUTPUT_PTR, \$IOB\$OUTPUT+4		
		2244		6E	9E	000F7			MOVAB	\$IOB\$OUTPUT, IOB\$+68		
		222C		07	90	000FC			MOVB	#7, IOB\$+44		
				EF	9F	00101	00000000G		PUSHAB	XPOSFAILURE		
				7E	D4	00107			CLRL	-(SP)		
				CB	9F	00109	2200		PUSHAB	IOB\$		
		00000000G		03	FB	0010D			CALLS	#3, XPOSPUT		
		2244		CB	9E	00114	232C		MOVAB	\$IOB\$OUTPUT, IOB\$+68		0364
		222C		07	90	0011B			MOVB	#7, IOB\$+44		
				EF	9F	00120	00000000G		PUSHAB	XPOSFAILURE		
				7E	D4	00126			CLRL	-(SP)		
				CB	9F	00128	2200		PUSHAB	IOB\$		
		00000000G		03	FB	0012C			CALLS	#3, XPOSPUT		
				8F	9A	00133	82		MOVZBL	#130, R8		0369

58	00	0000'	57 CF	68 15 67 2A 15 58 6A	D0 2C 18 C0 C2 2C	00137 0013A 00141 00142 00144 00147 0014A	MOVL MOVCS BGEQ ADDL2 SUBL2 MOVCS	OUTPUT_PTR, R7 #21, P.AAU, #0, R8, (R7) 12\$ #21, R7 #21, R8 #3, (PREFIX_PTR), #0, R8, (R7)	0377	
58	00		57 58 69	03 67 1C 03 03 56 67	C0 C2 2C 18 C0 C2 2C	00152 00155 00158 0015D 0015E 00160 00163	BGEQ ADDL2 SUBL2 MOVCS BGEQ ADDL2 SUBL2	12\$ #3, R7 #3, R8 MACRO_LENGTH, (MACRO_PTR), #0, R8, (R7) 12\$ MACRO_LENGTH, R7 MACRO_LENGTH, R8		
58	00	0000'	57 58 CF	56 21 67 00 02 51	C0 C2 2C 3A 12 D4	00166 0016D 0016E 00176 00178	MOVCS LOCC BNEQ CLRL	#33, P.AAV, #0, R8, (R7) #0, #131, OUTPUT_RECORD 13\$ R1	0378	
	FF7C	CB	0083	8F						
		6E		50 51 02 03 04 2244 222C	FF7C CB 9E A3 90 90 D0 9E 90 9F D4 9F FB 9E D0 2C	0017A 0017F 00183 00187 0018B 0018F 00194 00199 0019F 001A1 001A5 001AC 001B0 001B3 001BA 001BB 001BD 001C0 001C3 001C8 001C9 001CB 001CE 001D1 001D6 001D7 001D9 001DC 001DF 001E6 001E7 001EB 001EF 001F1 001F6 001F9	13\$: MOVAB SUBW3 MOVB MOVB MOVL MOVAB MOVB PUSHAB CLRL PUSHAB CALLS MOVAB MOVL MOVCS	OUTPUT_RECORD, R0 R0, R1, \$IOBSOUTPUT #14, \$IOBSOUTPUT+2 #1, \$IOBSOUTPUT+3 OUTPUT_PTR, \$IOBSOUTPUT+4 \$IOBSOUTPUT, IOBS+68 #7, IOBS+44 XPOSFAILURE -(SP) IOBS #3, XPOSPUT 8(R6), R8 OUTPUT_PTR, R7 #4, P.AAV, #0, R8, (R7)	0390 0391	
58	00	0000'	57 58 CF	04 67 2A 04 04 03 67 1C 03 03 56 67 0E 56 56 01 67	C0 2C 18 C0 C2 2C 18 C0 C2 2C 18 C0 C2 2C	001B3 001BA 001BB 001BD 001C0 001C3 001C8 001C9 001CB 001CE 001D1 001D6 001D7 001D9 001DC 001DF	BGEQ ADDL2 SUBL2 MOVCS BGEQ ADDL2 SUBL2 MOVCS BGEQ ADDL2 SUBL2 MOVCS	14\$ #4, R7 #4, R8 #3, (PREFIX_PTR), #0, R8, (R7) 14\$ #3, R7 #3, R8 MACRO_LENGTH, (MACRO_PTR), #0, R8, (R7) 14\$ MACRO_LENGTH, R7 MACRO_LENGTH, R8		
58	00	0000'	57 58 CF	01 67 01 53 03 50 83	C0 2C DD DD DD FB D0 2C	001E6 001E7 001EB 001EF 001F1 001F6 001F9	MOVCS PUSHL PUSHL PUSHL CALLS MOVL MOVW	#1, P.AAX, #0, R8, (R7) CALL_PTR CALL_LENGTH PTR #3, OUTLIST R0, PTR F.AAY, (PTR)+	0393 0394	

EXI
Mo
--
LA
LP
RM
SY
SY


```

407 0404 1 ROUTINE OUTLIST(BUFPTR, LEN, LPTR)=
408 0405 1
409 0406 1 *
410 0407 1 FUNCTION
411 0408 1 Dump an argument list buffer (which may span several source
412 0409 1 records..) to the output file. Continuation lines are flagged
413 0410 1 by an ASCII NUL (0) byte.
414 0411 1 INPUTS
415 0412 1 BUFPTR - byte-pointer into the OUTPUT_RECORD containing prefix
416 0413 1 information for the first record written.
417 0414 1 LEN - Length of the argument-list string.
418 0415 1 LPTR - Pointer to the argument-list string.
419 0416 1 OUTPUT
420 0417 1 Updated 'BUFPTR'
421 0418 1
422 0419 1 -
423 0420 2 BEGIN
424 0421 2 LOCAL
425 0422 2 IPTR, : Input string pointer (arg-list)
426 0423 2 OPTR, : Copy of BUFPTR...
427 0424 2 CHAR; : Character temporary
428 0425 2
429 0426 2 IPTR = .LPTR;
430 0427 2 OPTR = .BUFPTR;
431 0428 2
432 0429 2 DECR I FROM .LEN-1 TO 0 DO
433 0430 3 BEGIN
434 0431 3 CHAR = CH$RCHAR A( IPTR ); : Get next character and copy to
435 0432 3 CH$WCHAR_A( .CHAR, OPTR ); : output string
436 0433 3
437 0434 3 IF .CHAR EQL 0
438 0435 3 THEN
439 0436 4 BEGIN
440 0437 4 $XPO_PUT( IOB=OUTPUT,
441 0438 4 STRING=( TRUNCATED_OUTPUT, .OUTPUT_PTR)
442 0439 4 );
443 0440 4 OPTR = .OUTPUT_PTR; : Restore buffer ptr.
444 0441 4 CH$WCHAR_A( %CRAR(9), OPTR ) : and TAB from left-margin
445 0442 4 END
446 0443 2 END;
447 0444 2
448 0445 2 .OPTR
449 0446 1 END;

```

IOB\$=

OUTPUT

		003C 0000	OUTLIST: .WORD	Save R2,R3,R4,R5	: 0404
5E		08 C2 0002	SUBL2	#8, SP	...
55	0C	AC D0 0005	MOVL	LPTR, IPTR	: 0426
52	04	AC 7D 0009	MOVQ	BUFPTR, OPTR	: 0427
		52 11 0000	BRB	3\$: 0429
54		85 9A 000F	1\$: MOVZBL	(IPTR)+, CHAR	: 0431
82		54 90 0012	MOVB	CHAR, (OPTR)+	: 0432
		54 D5 0015	TSTL	CHAR	: 0434

0000'	CF	0083	8F		48 12 00017	BNEQ	3\$		
					00 3A 00019	LOCC	#0, #131, OUTPUT_RECORD		0439
					02 12 00021	BNEQ	2\$		
					51 D4 00023	CLRL	R1		
			50	0000'	CF 9E 00025	MOVAB	OUTPUT_RECORD, R0		
6E			51		50 A3 0002A	SUBW3	R0, R1, \$IOB\$OUTPUT		
		02	AE		0E 90 0002E	MOVB	#14, \$IOB\$OUTPUT+2		
		03	AE		01 90 00032	MOVB	#1, \$IOB\$OUTPUT+3		
		04	AE	0000'	CF D0 00036	MOVL	OUTPUT_PTR, \$IOB\$OUTPUT+4		
		0000'	CF		6E 9E 0003C	MOVAB	\$IOB\$OUTPUT, ICB\$+68		
		0000'	CF		07 90 00041	MOVB	#7, IOB\$+44		
				00000000G	EF 9F 00046	PUSHAB	XPOSFAILURE		
					7E D4 0004C	CLRL	-(SP)		
					CF 9F 0004E	PUSHAB	IOB\$		
				00000000G	03 FB 00052	CALLS	#3, XPOSPUT		
					CF D0 00059	MOVL	OUTPUT_PTR, OPTR		0440
					09 90 0005E	MOVB	#9, (OPTR)+		C441
					53 F4 00061	SOBGEQ	I, 1\$		0434
					52 D0 00064	MOVL	OPTR, R0		0446
					04 00067	RET			

: Routine Size: 104 bytes, Routine Base: \$CODE\$ + 04B9

Sy
-S
L P
RM
SS
ST
SY
SY
SY
SY
SY
SY
SY

```

451 0447 1 ROUTINE CONVARGLIST(TEMP_PTR): NOVALUE=
452 0448 1
453 0449 1 :++
454 0450 1 : FUNCTIONAL DESCRIPTION:
455 0451 1 : This routine gathers the argument list of a macro definition.
456 0452 1 : It processes continuation lines. It eliminates unnecessary
457 0453 1 : characters and places the arguments with default values in ARG_LIST
458 0454 1 : and a list of the formal names in CALL_LIST
459 0455 1
460 0456 1 : PARAMETERS:
461 0457 1
462 0458 1 :     TEMP_PTR           = The value of a CH pointer in the macro
463 0459 1 :                       definition line.
464 0460 1
465 0461 1 : IMPLICIT OUTPUTS:
466 0462 1 :     ARG_LIST           = String of formal arguments with default values
467 0463 1 :     ARG_PTR            = String pointer to ARG_LIST
468 0464 1 :     ARG_LENGTH         = Length of string in ARG_LIST
469 0465 1 :     CALL_LIST          = String of formal arguments
470 0466 1 :     CALL_PTR           = String pointer to CALL_LIST
471 0467 1 :     CALL_LENGTH        = Length of string in CALL_LIST
472 0468 1 :     VAR_ARGS           = True if the last formal argument was END_VARNUM_ARGS
473 0469 1
474 0470 1 : ROUTINE VALUE:
475 0471 1 :     NONE
476 0472 1
477 0473 1 :--
478 0474 1
479 0475 2 : BEGIN
480 0476 2 : LOCAL
481 0477 2 :     LIST_PTR,          : Input character string pointer
482 0478 2 :     CHAR,              : Character from string
483 0479 2 :     STATE,             : State of parse
484 0480 2 :                       0 = No argument seen yet
485 0481 2 :                       1 = Scanning formal name
486 0482 2 :                       2 = Scanning default value
487 0483 2 :     BEGIN_ARG;        : Character string pointer to beginning of argument
488 0484 2
489 0485 2 : MACRO
490 M 0486 2 :     TRY_ADD_DEFAULT=
491 0487 2 :     IF STATE EQL 1 THEN (CH$WCHAR_A(%C=' ', ARG_PTR); CH$WCHAR_A(%C'0', ARG_PTR)) %;
492 0488 2
493 0489 2
494 0490 2 : Initialize
495 0491 2 :
496 0492 2 : ARG_PTR = CH$PTR(ARG_LIST);
497 0493 2 : CH$FILL(0, MAX ARG_LIST, ARG_PTR);
498 0494 2 : CALL_PTR = CH$PTR(CALL_LIST);
499 0495 2 : CH$FILL(0, MAX ARG_LIST, CALL_PTR);
500 0496 2 : LIST_PTR = CH$PLUST(TEMP_PTR, 3);
501 0497 2 : VAR_ARGS = 0;
502 0498 2 : STATE = 0;
503 0499 2
504 0500 2
505 0501 2
506 0502 2
507 0503 2 : Scan the argument list

```

Val

000
000
000
000
000
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF

```

508 0504 2 ! Exit when an end of line (not preceded by line continuation mark) is read
509 0505 2 !
510 0506 2 REPEAT
511 0507 2 BEGIN
512 0508 2 CHAR = CHSRCHAR_A(LIST_PTR);
513 0509 2 SELECTIONE .CHAR OF
514 0510 2 SET
515 0511 2
516 0512 2 [0]:
517 0513 2 BEGIN
518 0514 2 End of list
519 0515 2 Add a default value to the last argument if it did not
520 0516 2 have one.
521 0517 2 Set VAR_ARGS if the name of the last argument is
522 0518 2 END_VARNUM_ARGS
523 0519 2
524 0520 2 TRY ADD DEFAULT;
525 0521 2 IF .STATE GEQ 1
526 0522 2 THEN
527 0523 2 IF
528 0524 2 CHSEQ(
529 0525 2 CHSDIFF(.CALL_PTR, .BEGIN_ARG),
530 0526 2 .BEGIN_ARG,
531 0527 2 CH$LEN_PTR('END_VARNUM_ARGS'),
532 0528 2 0)
533 0529 2 THEN
534 0530 2 VAR_ARGS = 1;
535 0531 2
536 0532 2 EXITLOOP
537 0533 2 END;
538 0534 2
539 0535 2 [XC'A' TO XC'Z', XC'0' TO XC'9', XC'_' , XC'$']:
540 0536 2 BEGIN
541 0537 2 Symbol constituent
542 0538 2 If STATE is zero, it is the first character of a formal name
543 0539 2 STATE distinguishes scanning formal name from scanning default value
544 0540 2
545 0541 2 IF .STATE EQL 0
546 0542 2 THEN
547 0543 2 BEGIN
548 0544 2 STATE = 1;
549 0545 2 BEGIN_ARG = .CALL_PTR
550 0546 2 END;
551 0547 2
552 0548 2 IF .STATE EQL 1 THEN CH$WCHAR_A(.CHAR, CALL_PTR);
553 0549 2 CH$WCHAR_A(.CHAR, ARG_PTR)
554 0550 2 END;
555 0551 2
556 0552 2 [XC'=']:
557 0553 2 BEGIN
558 0554 2 Beginning of default value
559 0555 2 Set STATE to indicate scanning default and collect character
560 0556 2
561 0557 2 STATE = 2;
562 0558 2
563 0559 2
564 0560 2

```

Vi
St
IM
IM
IM
NU
NU
NU
NU
NU
Us
IM
Ma
Es

Pe
--

To
Us
To
Nu
9
A
LI


```

: 565      0561 4      CH$WCHAR_A(.CHAR, ARG_PTR)
: 566      0562 3      END;
: 567      0563 3
: 568      0564 3      [XC' ']:
: 569      0565 4      BEGIN
: 570      0566 4
: 571      0567 4      End of argument
: 572      0568 4      Add a default value to the argument if it did not have one
: 573      0569 4      Collect the character
: 574      0570 4      Reset STATE to indicate not within an argument
: 575      0571 4
: 576      0572 4      TRY ADD DEFAULT;
: 577      0573 4      CH$WCHAR_A(.CHAR, CALL_PTR);
: 578      0574 4      CH$WCHAR_A(.CHAR, ARG_PTR);
: 579      0575 4      STATE = 0
: 580      0576 3      END;
: 581      0577 3
: 582      0578 3      [XC' -']:
: 583      0579 4      BEGIN
: 584      0580 4
: 585      0581 4      Line continuation indicator
: 586      0582 4      Get a new line and reset LIST_PTR
: 587      0583 4
: 588      0584 4      CH$WCHAR_A( 0, CALL_PTR );
: 589      0585 4      CH$WCHAR_A( 0, ARG_PTR );
: 590      0586 4
: 591      0587 4      $XPO GET(IOB=INPUT);
: 592      0588 4      INPUT_LENGTH = .INPUT[IOB$H STRING];
: 593      0589 4      INPUT_PTR = .INPUT[IOB$A STRING];
: 594      0590 4      CH$COPY(.INPUT_LENGTH, .INPUT_PTR,
: 595      0591 4      0,
: 596      0592 4      MAX_REC_SIZE, .OUTPUT_PTR);
: 597      0593 4      LIST_PTR = .OUTPUT_PTR
: 598      0594 4      END;
: 599      0595 3
: 600      0596 3      [XC' #']:
: 601      0597 3      MACRO comment character
: 602      0598 3      The remainder of the line is "ignorable"
: 603      0599 3
: 604      0600 3
: 605      0601 3      EXITLOOP
: 606      0602 3      TES
: 607      0603 2      END;
: 608      0604 2
: 609      0605 2
: 610      0606 2      ! Set up implicit outputs as noted
: 611      0607 2      ! If variable number of arguments, then replace CALL_LIST by '%REMAINING'
: 612      0608 2      ! since this is the call argument
: 613      0609 2
: 614      0610 2      ARG_LENGTH = CH$DIFF(.ARG_PTR, CH$PTR(ARG_LIST));
: 615      0611 2      ARG_PTR = CH$PTR(ARG_LIST);
: 616      0612 2
: 617      0613 2      IF .VAR_ARGS
: 618      0614 2      THEN
: 619      0615 2      BEGIN
: 620      0616 2      CH$COPY(
: 621      0617 2      CH$LEN_PTR('%REMAINING'),

```

```

: 622      0618      3
: 623      0619      3
: 624      0620      3
: 625      0621      3
: 626      0622      3
: 627      0623      3
: 628      0624      3
: 629      0625      3
: 630      0626      3
: 631      0627      1
:
: INFO#250      L1:0526
: Referenced LOCAL symbol BEGIN_ARG is probably not initialized

```

```

0,
MAX_ARG_LIST,
CHSPTR(CALL_LIST));
CALL_LENGTH = %CHARCOUNT('%REMAINING');
END
ELSE
CALL_LENGTH = CHSDIFF(.CALL_PTR, CHSPTR(CALL_LIST));
CALL_PTR = CHSPTR(CALL_LIST)
END;

```

Mo
LA
LA
LA
LA
SY
SY

														.PSECT \$SPLITS,NOWRT,NOEXE,2							
53	47	52	41	5F	4D	55	4E	52	41	56	5F	44	4E	45	00114	P.ABA:	.ASCII	\END_VARNUM_ARGS\<0>	:		
															00	00123			:		
				00	00	47	4E	49	4E	49	41	4D	45	52	25	00124	P.ABB:	.ASCII	\%REMAINING\<0><0>	:	
														IOB\$=		INPUT					
														.PSECT \$CODE\$,NOWRT,2							
														07FC 0000 CONVARGLIST:							
														.WORD		Save R2,R3,R4,R5,R6,R7,R8,R9,R10		: 0447			
														MOVAB		ARG_PTR, R10		: 0493			
OFFF	8F			00																:	
														MOVAB		ARG_LIST, ARG_PTR		: 0494			
														MOVCS		#0, (SP), #0, #4095, @ARG_PTR		: 0495			
OFFF	8F			00	1008	CA	1010	CA	9E	00014										:	
														MOVAB		CALL_LIST, CALL_PTR		: 0496			
														MOVCS		#0, (SP), #0, #4095, @CALL_PTR		: 0497			
														ADDL3		#3, TEMP_PTR, LIST_PTR		: 0498			
														CLRL		VAR_ARGS		: 0499			
														CLRL		STATE		: 0508			
														MOVZBL		(LIST_PTR)+, CHAR		: 0512			
														BNEQ		5\$: 0513			
														CMPL		STATE, #1					
														BNEQ		3\$					
														MOVB		#61, @ARG_PTR					
														INCL		ARG_PTR					
														MOVB		#48, @ARG_PTR					
														INCL		ARG_PTR					
														TSTL		STATE		: 0522			
														BLEQ		4\$					
OF				50	1008	CA														:	
														SUBL3		BEGIN_ARG, CALL_PTR, R0		: 0526			
														CMPCS		R0, (BEGIN_ARG), #0, #15, P.ABA					
														CF		00055					
														BNEQ		4\$					
														MOVL		#1, VAR_ARGS		: 0531			
														BRW		17\$: 0513			
														CMPL		CHAR, #36		: 0536			
														BEQL		8\$					
														CMPL		CHAR, #48					

			05	19	0006A	BLSS	6\$				
		39	56	D1	0006C	CMPL	CHAR, #57				
			18	15	0006F	BLEQ	8\$				
		00000041	8F	56	D1	00071	6\$: CMPL	CHAR, #65			
			09	19	00078	BLSS	7\$				
		0000005A	8F	56	D1	0007A	CMPL	CHAR, #90			
			09	15	00081	BLEQ	8\$				
		0000005F	8F	56	D1	00083	7\$: CMPL	CHAR, #95			
			1C	12	0008A	BNEQ	10\$				
			57	D5	0008C	8\$: TSTL	STATE			0543	
			08	12	0008E	BNEQ	9\$				
		57	01	D0	00C90	MOVL	#1, STATE			0546	
		58	3008	CA	D0	00093	MOVL	CALL_PTR, BEGIN_ARG		0547	
		01	57	D1	00098	9\$: CMPL	STATE, #1			0550	
			13	12	0009B	BNEQ	11\$				
		1008	DA	56	90	0009D	MOVB	CHAR, @CALL_PTR			
			3008	CA	D6	000A2	INCL	CALL_PTR			
			08	11	000A6	BRB	11\$			0551	
		3D	56	D1	000AB	10\$: CMPL	CHAR, #61			0554	
			0B	12	000AB	BNEQ	12\$				
		57	02	D0	000AD	MOVL	#2, STATE			0560	
		00	BA	56	90	000B0	11\$: MOVB	CHAR, @ARG_PTR		0561	
			6A	D6	000B4	INCL	ARG_PTR				
			71	11	000B6	BRB	15\$			0509	
		2C	56	D1	000B8	12\$: CMPL	CHAR, #44			0564	
			23	12	000BB	BNEQ	14\$				
		01	57	D1	000BD	CMPL	STATE, #1			0565	
			0C	12	000C0	BNEQ	13\$				
		00	BA	3D	90	000C2	MOVB	#61, @ARG_PTR			
			6A	D6	000C6	INCL	ARG_PTR				
		00	BA	30	90	000C8	MOVB	#48, @ARG_PTR			
			6A	D6	000CC	INCL	ARG_PTR				
		1008	DA	56	90	000CE	13\$: MOVB	CHAR, @CALL_PTR		0573	
			3008	CA	D6	000D3	INCL	CALL_PTR			
		00	BA	56	90	000D7	MOVB	CHAR, @ARG_PTR		0574	
			6A	D6	000DB	INCL	ARG_PTR				
			FF4E	31	000DD	BRW	1\$			0575	
		2D	56	D1	000E0	14\$: CMPL	CHAR, #45			0578	
			47	12	000E3	BNEQ	16\$				
			3008	DA	94	000E5	CLRB	@CALL_PTR		0584	
			3008	CA	D6	000E9	INCL	CALL_PTR			
			00	BA	94	000ED	CLRB	@ARG_PTR		0585	
			6A	D6	000F0	INCL	ARG_PTR				
		2134	CA	06	90	000F2	MOVB	#6, -IOB\$+44		0587	
			00000000G	EF	9F	000F7	PUSHAB	XPOSFAILURE			
				7E	D4	000FD	CLRL	-(SP)			
			2108	CA	9F	000FF	PUSHAB	IOB\$			
		00000000G	EF	03	FB	00103	CALLS	#3, XPOSGET			
		FF74	CA	213C	CA	3C	0010A	MOVZWL	INPUT+52, INPUT_LENGTH	0588	
		FF70	CA	2140	CA	D0	00111	MOVL	INPUT+56, INPUT_PTR	0589	
0082	8F	00	FF70	DA	FF74	CA	2C	00118	MOVCS	INPUT_LENGTH, @INPUT_PTR, #0, #130, -	0592
			FC	BA				00123	@OUTPUT_PTR		
		59	FC	AA	D0	00125	MOVL	OUTPUT_PTR, LIST_PTR		0593	
			FF04	31	00129	15\$: BRW	2\$				
		3B	56	D1	0012C	16\$: CMPL	CHAR, #59			0596	
			F8	12	0012F	BNEQ	15\$				
		50	08	AA	9E	0C131	17\$: MOVAB	ARG_LIST, R0		0610	

	04	AA		6A		50	C3	00135		SUBL3	R0, ARG_PTR, ARG_LENGTH	:	
				6A	08	AA	9E	0013A		MOVAB	ARG_LIST, ARG_PTR	:	0611
OFFF	8F			13	2010	CA	E9	0013E		BLBC	VAR_ARGS, 18\$:	0613
		00	0000'	CF		0A	2C	00143		MOVCS	#10, P.ABB, #0, #4095, CALL_LIST	:	0620
			100C	CA	1010	CA		0014C				:	
						0A	D0	0014F		MOVL	#10, CALL_LENGTH	:	0621
						0D	11	00154		BRB	19\$:	0613
				50	1010	CA	9E	00156	18\$:	MOVAB	CALL_LIST, R0	:	0624
100C	CA		1008	CA		50	C3	0015B		SUBL3	R0, CALL_PTR, CALL_LENGTH	:	
			1008	CA	1010	CA	9E	00163	19\$:	MOVAB	CALL_LIST, CALL_PTR	:	0626
						04		0016A		RET		:	0627

; Routine Size: 363 bytes, Routine Base: \$CODE\$ + 0521

This image displays a dense grid of technical diagrams and code snippets, likely from a manual or reference document. The diagrams are arranged in a grid pattern, with some larger diagrams interspersed among smaller ones. The diagrams include various types of maps, lists, and code blocks. Key labels visible in the grid include:

- MELDR
- LALOAD MAP
- XFLORDER MAP
- LADAMCODE LIS
- LALOAD LIS
- LAMRMCODE LIS
- LADMT LIS
- MARBLI
- MARBLI MAP
- MARBLI LIS
- MDL32 MAP

The diagrams themselves consist of vertical columns of text, often with horizontal lines or boxes, suggesting structured data or code listings. The overall appearance is that of a technical reference or a collection of code examples for a specific system or language.