


```

IIIIII  NN      NN  PPPPPPP  UU      UU  TTTTTTTTTT
IIIIII  NN      NN  PPPPPPP  UU      UU  TTTTTTTTTT
  II    NN      NN  PP        PP  UU      UU  TT
  II    NN      NN  PP        PP  UU      UU  TT
  II    NNNN    NN  PP        PP  UU      UU  TT
  II    NNNN    NN  PP        PP  UU      UU  TT
  II    NN  NN  NN  PPPPPPP  UU      UU  TT
  II    NN  NN  NN  PPPPPPP  UU      UU  TT
  II    NN      NNNN  PP      UU      UU  TT
  II    NN      NNNN  PP      UU      UU  TT
  II    NN      NN  PP      UU      UU  TT
  II    NN      NN  PP      UU      UU  TT
  II    NN      NN  PP      UU      UU  TT
IIIIII  NN      NN  PP      UU      UU  TT
IIIIII  NN      NN  PP      UUUUUUUUUU  TT
IIIIII  NN      NN  PP      UUUUUUUUUU  TT

```

```

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
LLLLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLLLL  IIIIII  SSSSSSSS

```



```

0000 1 .TITLE MAC$INPUT GET NEXT CHARACTER
0000 2 .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6 *
0000 7 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 * ALL RIGHTS RESERVED.
0000 10 *
0000 11 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 * TRANSFERRED.
0000 17 *
0000 18 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 * CORPORATION.
0000 21 *
0000 22 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 *
0000 25 *
0000 26 *****
0000 27
0000 28
0000 29 ++
0000 30 FACILITY: VAX MACRO ASSEMBLER OBJECT LIBRARY
0000 31
0000 32 ABSTRACT:
0000 33
0000 34 The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000 35 modules for input to the VAX-11 LINKER.
0000 36
0000 37 ENVIRONMENT: USER MODE
0000 38
0000 39 AUTHOR: Benn Schreiber, CREATION DATE: 21-AUG-78
0000 40
0000 41 MODIFIED BY:
0000 42
0000 43 V03.01 MTR0011 Mike Rhodes 18-Mar-1982
0000 44 Add logic to routine MAC$GETLIN to trap non-RM$$ EOF errors
0000 45 and non-SUM$ xxx errors. This occurs when an illegal record
0000 46 is encountered by RMS and it does not count as a SUM error,
0000 47 which leaves us in an infinite loop trying to read the next line.
0000 48 Fixes QAR #691.
0000 49
0000 50 V03.00 MTR0006 Mike Rhodes 15-Mar-1982
0000 51 Modify MAC$GETCHR to allow ALL characters to be passed
0000 52 when the FLG$V DLIMSTR flag is set. The characters allowed
0000 53 to be passed include the semicolon (which was already passed)
0000 54 and the hyphen, which at times was incorrectly treated as a
0000 55 line continuation character. Fixes SPR #11-42904.
0000 56
0000 57 V01.10 RN0023 R. Newland 3-Nov-1979

```

0000 58 :
0000 59 :
0000 60 :
0000 61 :
0000 62 :
0000 63 :
0000 64 :
0000 65 :
0000 66 :
0000 67 :
0000 68 :
0000 69 :
0000 70 :
0000 71 :--

New message codes to get error messages from system message file.

V01.09 RN0010 R. Newland 5-Sep-1979
Multipage MXB blocks

V01.08 RN0005 R. Newland 27-Aug-1979
Remove .ALIGN LONG statements

V01.07 RNG002 R. Newland 01-Feb-1979
Changes for Source Update Merge, Get input lines from SUM\$LINE.

```
0000 73          .SBTTL DECLARATIONS
0000 74          :
0000 75          : INCLUDE FILES:
0000 76          :
0000 77          :
0000 78          :
0000 79          : MACROS:
0000 80          :
0000 81          $RABDEF          ;DEFINE RAB OFFSETS
0000 82          $MAC_GENVALDEF   ;VAX-11 MACRO GENERAL SYMBOLS
0000 83          $MAC_CTLFLGDEF   ;CONTROL FLAGS
0000 84          $MAC_INTCODDEF   ;INTERMEDIATE CODE FILE SYMBOLS
0000 85          $MAC_INPBLKDEF   ;DEFINE INPUT BLOCK OFFSETS
003C 86          $MAC_SYMBLKDEF   ;DEFINE SYMBOL BLOCK OFFSETS
0000 87          $MAC_MNBDEF      ;DEFINE MNB OFFSETS
0000 88          $MACMSGDEF       ; Define message codes
0000 89          $STSDEF          ;STATUS BITS
0000 90          DEFSUMCBL        ; Define SUM control block symbols
0000 91          :
0000 92          :
0000 93          :
0000 94          : EQUATED SYMBOLS:
0000 95          :
0000 96          :
0000 97          :
0000 98          : OWN STORAGE:
0000 99          :
0000 100         :
```

```

0000 102      .SBTTL  MAC$GETCHR GET NEXT CHARACTER FROM INPUT STREAM
0000 103      :++
0000 104      : FUNCTIONAL DESCRIPTION:
0000 105      :
0000 106      :     THIS ROUTINE IS CALLED WHENEVER A NEW CHARACTER IS NEEDED.
0000 107      :     IT PERFORMS ALL THE HOUSEKEEPING FOR ENDS OF LINES, OUTPUTS
0000 108      :     COMMANDS TO THE INT. FILE FOR NEW LINES, AND HANDLES CONTINUATION
0000 109      :     LINES.
0000 110      :
0000 111      : CALLING SEQUENCE:
0000 112      :
0000 113      :     JSB      MAC$GETCHR
0000 114      :
0000 115      : INPUT PARAMETERS:
0000 116      :
0000 117      :     R10     LAST CHARACTER READ
0000 118      :     R11     POINTS TO FLAGS (MAC$GL_FLAGS)
0000 119      :     IF FLG$M ALLCHR IS SET THEN SEMICOLONS WILL BE PASSED
0000 120      :     BACK. IF THE FLAG IS CLEAR, SEMICOLONS AND
0000 121      :     EVERYTHING FOLLOWING WILL BE IGNORED.
0000 122      :
0000 123      : IMPLICIT INPUTS:
0000 124      :
0000 125      :     NONE
0000 126      :
0000 127      : OUTPUT PARAMETERS:
0000 128      :
0000 129      :     R10     NEXT CHARACTER
0000 130      :
0000 131      : IMPLICIT OUTPUTS:
0000 132      :
0000 133      :     NONE
0000 134      :
0000 135      : COMPLETION CODES:
0000 136      :
0000 137      :     NONE
0000 138      :
0000 139      : SIDE EFFECTS:
0000 140      :
0000 141      :     NONE
0000 142      :
0000 143      :--
0000 144      :
0000 145      :
0000 146      :
00000000 147      .PSECT  MAC$RO_CODE_P15,NOWRT,GBL,LONG
0000 148      :
0000 149      .ENABL  LSB
0000 150      :
0000 151      MAC$GETCHR::
58      0000'58  DD      0000 152      PUSHL   R8           ;SAVE R8
0D      0000'CF  9E      0002 153 10$:   MOVAB   W^MAC$GL_LINEPT,R8 ;POINT TO MAC$GL_LINEPT
1F      0000'5A  91      0007 154 20$:   CMPB   R10,#CR          ;TIME TO READ NEW LINE?
5A      0000'20  9A      000C 155      BNEQ   30$           ;IF NEQ NO
6B      0000'CF  D0      000F 156 25$:   MOVZBL #BLNK,R10       ;YES--PREVENT LOOPING ON CR
0000'08  C8      0014 157      MOVL   W^MAC$GL_INPUTP,R0 ;ADDRESS INPUT BLOCK
0000'08  C8      0014 158      BISL2  #FLG$M_CONT,(R11) ;ALLOW CONTINUATION LINES

```

```

08 B0 16 0017 159 JSB @INP$L_GETL(R0) ;CALL INPUT ROUTINE TO READ NEXT LINE
E9 6B 10 E0 001A 160 BBS #FLG$V_MACTXT,(R11),20$ ;BRANCH IF READING MACRO TEXT
001E 161 $INTOUT_WD INT$ _NEWL,- ;YES--SIGNAL NEW LINE FOR PASS2
001E 162 <W^MAC$GL_LINENUM>
5A 00 DC 11 0029 163 BRB 20$ ;CONTINUE
00 B8 9A 002B 164 30$: MOVZBL @(R8),R10 ;GET NEXT CHARACTER
68 D6 002F 165 INCL (R8) ;POINT TO NEXT CHARACTER
5A D5 0031 166 TSTL R10 ;WAS CHARACTER A NULL?
F6 13 0033 167 BEQL 30$ ;IF EQL YES--GET NEXT CHARACTER
0035 168 ;(GOD ONLY KNOWS WHERE NULLS COME FROM!)
03 6B 2F E1 0035 169 BBC #FLG$V_DLIMSTR,(R11),35$ ;ALLOW SEMICOLONS AND HYPHENS?
007B 31 0039 170 BRW 150$ ;YES -- BYPASS OTHER TESTS.
0E 6B 1A E0 003C 171 35$: BBS #FLG$V_SPLALL,(R11),40$ ;BRANCH IF SHOULD NOT EVEN CONSIDER
0040 172 ;SEMI-COLONS
3B 5A 91 0040 173 CMPB R10,#SEMI ;IS CHARACTER A SEMI-COLON?
09 12 0043 174 BNEQ 40$ ;IF NEQ NO
06 6B E8 0045 175 BLBS (R11),40$ ;YES--AND ARE WE PASSING ALL CHARS.?
0048 176 ;(ALLCHR IS LOW BIT!!)
5A 0D 9A 0048 177 MOVZBL #CR,R10 ;NO--CALL IT EOL
0069 31 004B 178 BRW 150$ ;TAKE THE QUICK EXIT
2D 5A 91 004E 179 40$: CMPB R10,#HYPHEN ;LINE CONTINUATION?
64 12 0051 180 BNEQ 150$ ;IF NEQ NO
60 6B 03 E1 0053 181 BBC #FLG$V_CONT,(R11),150$ ;YES--CONTINUATIONS ALLOWED?
5A DD 0057 182 PUSHL R10 ;YES--SAVE CURRENT STATE
68 DD 0059 183 PUSHL (R8) ;SAVE MAC$GL_LINEPT
7E 6B FFFFFFFE 8F CB 005B 184 BICL3 #^C<FLG$M_ALLCHR>,(R11),-(SP) ;SAVE ALLCHR STATE
68 09 CA 0063 185 BICL2 #FLG$M_CONT!FLG$M_ALLCHR,(R11) ;DON'T RECURSE ON LINES THAT HAVE
5A 20 9A 0066 186 MOVZBL #BLNK,R10 ; ALL HYPHENS.
20 5A 91 0069 187 60$: CMPB R10,#BLNK ;IS CHARACTER A BLANK?
05 13 006C 188 BEQL 70$ ;IF EQL YES
09 5A 91 006E 189 CMPB R10,#TAB ;NO--IS IT A TAB?
05 12 0071 190 BNEQ 80$ ;IF NEQ NO
FF8A 30 0073 191 70$: BSBW MAC$GETCHR ;YES--SKIP OVER SPACES AND TABS
F1 11 0076 192 BRB 60$ ;FIND NON-BLANK, NON-TAB CHARACTER
0D 5A 91 0078 193 80$: CMPB R10,#CR ;IS THIS EOL?
2E 12 007B 194 BNEQ 130$ ;IF NEQ NO
68 8E C8 007D 195 BISL2 (SP)+,(R11) ;YES--RESTORE ALLCHR FLAG
5E 08 C0 0080 196 ADDL2 #2*4,SP ;AND SAVED CONTEXT NOT NEEDED
FF7A 30 0083 197 BSBW MAC$GETCHR ;READ NEXT LINE
04 6B 0D E1 0086 198 BBC #FLG$V_OPRND,(R11),90$ ;BRANCH IF NOT IN OPERAND FIELD
06 6B 14 E3 008A 199 BBS #FLG$V_CHKLPND,(R11),100$ ;SET CHKL PENDING AND BRANCH IF NONE PENDING
008E 200 90$: $INTOUT_X INT$ _CHKL ;ENSURE ALIGNMENT OF CONTINUED LINES
0D 5A 91 0094 201 100$: CMPB R10,#CR ;NULL LINE?
0D 13 0097 202 BEQL 120$ ;IF EQL YES
0C 5A 91 0099 203 CMPB R10,#FF ;STILL LOOKING FOR NULL LINES
05 13 009C 204 BEQL 110$
3B 5A 91 009E 205 CMPB R10,#SEMI
11 12 00A1 206 BNEQ 140$
5A 0D 9A 00A3 207 110$: MOVZBL #CR,R10 ;TREAT AS EOL IF NULL
FF57 30 00A6 208 120$: BSBW MAC$GETCHR ;READ NEXT LINE
E9 11 00A9 209 BRB 100$ ;FIND NON-NULL LINE
00AB 210 ;
00AB 211 ; NOT REALLY A CONTINUED LINE
00AB 212 ;
6B 8E C8 00AB 213 130$: BISL2 (SP)+,(R11) ;RESTORE ALLCHR FLAG
68 8ED0 00AE 214 POPL (R8) ;RESTORE MAC$GL_LINEPT
5A 8ED0 00B1 215 POPL R10 ;RESTORE CHARACTER

```


MACSINPUT
V04-000

G 16

GET NEXT CHARACTER 16-SEP-1984 02:20:18 VAX/VMS Macro V04-00
MACSGETCHR GET NEXT CHARACTER FROM INPUT 5-SEP-1984 01:48:32 [MACRO.SRC]INPUT.MAR;1

Page 6
(3)

```
6B 08 C8 00B4 216 140$: BISL2 #FLGSM_CONT,(R11) ;ALLOW CONTINUATIONS AGAIN
    58 8ED0 00B7 217 150$: POPL R8 ;RESTORE R8
      05 00BA 218 RSB ;RETURN WITH CHARACTER IN R10
      00BB 219
      00BB 220 .DSABL LSB
```

```

00BB 222      .SBTTL MAC$GETLIN GET NEXT INPUT SOURCE LINE
00BB 223
00BB 224      :++
00BB 225      :
00BB 226      : THIS ROUTINE IS CALLED TO GET THE NEXT LINE OF THE CURRENT
00BB 227      : INPUT FILE AND PLACE IT IN MAC$AB_LINEBF.
00BB 228      :--
00BB 229
00BB 230      .ENABL LSB
00BB 231
00BB 231 MAC$GETLIN::
50 0000'CF 9E 00BB 232 MOVAB W^MAC$INPUT_RAB,RO ;POINT TO THE RAB
00000000'GF 01 DF 00C0 233 PUSHAL W^MAC$GT_SCB ; Control block address
6D 50 E8 00CB 234 CALLS #1,G^SUM$LINE ; Get next input line
00000000'8F 50 D1 00CE 235 BLBS RO,40$ ; If LBS then good read
27 13 00D5 236 CMPL RO,#RMSS_EOF ; Was error end-of-file?
51 50 0C 10 EF 00D7 237 BEQL 8$ ; Yes if EQL, try next file
0084 8F 51 B1 00DC 238 EXTZV #STSSV FAC NO,#STSS$ FAC NO,RO,R1 ; Get facility no
OC 13 00E1 239 CMPW R1,#<SOMS_NORMAL@-16$ ; SUM error?
51 50 FFFFFFFF 8F CB 00E3 240 BEQL 2$ ; Yes if EQL
CE 13 00EB 241 BICL3 #^CSTSSM SEVERITY,RO,R1 ; Copy and check the severity
OF 11 00ED 242 BEQL MAC$GETLIN ; WARNING, try again
05 6B 0E E0 00EF 243 BRB 8$ ; ERROR or FATAL, try next file
00AF 30 00F3 244 2$: BBS #FLGSV P2,(R11),5$ ; Return line if Pass 2
C3 11 00F6 245 BSBW SUM_ERROR ; Generate intermediate code if Pass 1
00F8 246 BRB MAC$GETLIN
0000'CF D7 00F8 247 5$: DECL W^MAC$GL_LINENUM ; Don't increment line number
3D 11 00FC 248 BRB 40$
00F5 30 00FE 249 8$: BSBW MAC$NXTINPFIL ; ELSE TRY TO OPEN NEXT FILE
0000'CF 01 9A 0101 250 MOVZBL #1,W^MAC$GL_SRC PAG ; RESET PAGE COUNT TO ONE
B2 50 E8 0106 251 BLBS RO,MAC$GETLIN ; IF THERE IS ANOTHER FILE GO READ IT
0000'CF 444E452E 8F D0 0109 252 10$: MOVL #^A/.END/,W^MAC$AB_LINEBF ; OOPS--NO FILE--FAKE A '.END'
0022'CF 04 9B 0112 253 MOVZBW #4,W^MAC$INPUT_RAB+RAB$W_RSZ ; WITH A SIZE OF 4 BYTES
0000'CF D5 0117 254 TSTL W^MAC$GL_IF_LEVEL ; IN UNFINISHED CONDITIONAL?
OC 15 011B 255 BLEQ 30$ ; IF LEQ NO
0004'CF 43 8F 90 011D 256 MOVB #^A/C/,W^MAC$AB_LINEBF+4 ; YES--MAKE .END INTO .ENDC
0022'CF B6 0123 257 INCW W^MAC$INPUT_RAB+RAB$W_RSZ ; COUNT THE CHARACTER
OE 6B 0E E0 0127 258 BRB 40$ ; CONTINUE
0129 259 BBS #FLGSV P2,(R11),40$ ; ONLY MESSAGE ON PASS 1
012D 260 $INTOUT_X INT$ FNEWL ; PRINT CONTENTS OF PREVIOUS LINE
012D 261 $INTOUT_LW INT$ WRN,- ; TELL OF MISSING END STATEMENT
012D 262 <#MAC$_MISSINGEND,#0>
0022'CF 3C 013B 263 40$: MOVZWL W^MAC$INPUT_RAB+RAB$W_RSZ,- ; SAVE LENGTH OF LINE
50 0000'CF 50 D0 013F 264 RO
0000'CF 50 9E 0145 265 50$: MOVL RO,W^MAC$GL_LINELN ; :
0000'CF 50 D0 014A 266 MOVAB W^MAC$AB_LINEBF,RO ; GET ADDRESS OF LINE BUFFER
0000'CF 50 D0 014F 267 MOVL RO,W^MAC$GL_LINEPT ; SET UP LINE POINTER
OC 60 91 0154 268 MOVL RO,W^MAC$GL_ERRPTX ; RESET ERROR POINTER TO LINE START
04 12 0157 269 CMPB (RO),#FF ; NEW SOURCE PAGE?
0000'CF D6 0159 270 BNEQ 60$ ; IF NEQ NO
50 0000'CF C0 015D 271 INCL W^MAC$GL_SRC PAG ; YES--COUNT NEW PAGE
60 OD 90 0162 272 ADDL2 W^MAC$GL_LINELN,RO ; FIGURE ADDRESS OF LAST CHARACTER
0000'CF D6 0165 273 MOVB #CR,(RO) ; STORE CR FOR END OF LINE
04 6B 0E E0 0169 274 INCL W^MAC$GL_LINENUM ; COUNT THIS LINE
0000'CF D6 016D 275 BBS #FLGSV P2,(R11),80$ ; BRANCH IF THIS IS PASS 2
276 INCL W^MAC$GL_SRC_LCNT ; NO--COUNT LINE READ IN PASS 1
277
278

```

				0171	279	80\$:			
				0171	280		CLRL	R1	; Initialise insert number
				0173	281		BBS	#FLGSV UPDFIL,(R11),90\$; Branch if file is being updated
0000'CF	13 6B	27		0177	282		SUBL3	W^MAC\$GL_LINBAS,-	; Compute line number
				017E	283			W^MAC\$GL_LINENUM,R0	
				017F	284		BBC	#FLGSV SEQFIL,(R11),100\$; Branch if not sequenced file
50	17 6B	19		0183	285		MOVL	W^MAC\$GL_RECHDBUF,R0	; Get sequenced line number
				0188	286		BRB	100\$	
				018A	287	90\$:			
50	0018'CF	3C		018A	288		MOVZWL	W^MAC\$GT_SCB+SUM_W_LINE_NO,R0	; Get original line number
001C'CF	02			018F	289		BBC	#SUM_V_SRCUPD,-	; Branch if line from source file
				0194	290			W^MAC\$GT_SCB+SUM_B_FLAGS,100\$	
51	001A'CF	3C		0195	291		MOVZWL	W^MAC\$GT_SCB+SUM_W_INSERT_NO,R1	; Get insert number
				019A	292	100\$:			
0000'CF	50			019A	293		MOVW	R0,W^MAC\$GW_LST_LINE	; Save listing line number
0000'CF	51			019F	294		MOVW	R1,W^MAC\$GW_LST_INST	; and insert number
				01A4	295		RSB		
				01A5	296				
				01A5	297		.DSABL	LSB	
				01A5	298	:			
				01A5	299	:			
				01A5	300	SUM_ERROR:			
0000'CF	04	D5		01A5	301		TSTL	W^MAC\$GL_LINENUM	; Error on very first line?
				01A9	302		BNEQ	5\$; No if NEQ
00 6B	29	E2		01AB	303		BBSS	#FLGSV_FIRSTLN,(R11),5\$	
				01AF	304	5\$:			
00848800	8F	50		01AF	305		CMP	R0,#SUM\$_EDITSCLSH	; Edits clash error?
				01B6	306		BNEQ	4\$; No if NEQ
0000001C'EF	03	E0		01B8	307		BBS	#SUM_V_SUBCLSH,-	; Branch if not first clash
				01BF	308			MAC\$GT_SCB+SUM_B_FLAGS,10\$	
				01C0	309		PUSHL	R0	
				01C2	310		\$INTOUT_X	INTS_FNEWL	; Force new line
				01C8	311		POPL	R0	
50	03	8ED0		01CB	312		BISL2	#STSSK_INFO,R0	; Make and information code
				01CE	313		\$INTOUT_LW	INTS_INFO,<R0,#0>	; Information message
				01DB	314		BRB	-10\$	
				01DA	315	4\$:			
				01DA	316		PUSHL	R0	
				01DC	317		\$INTOUT_X	INTS_FNEWL	; Force new line
				01E2	318		POPL	R0	
				01E5	319		\$INTOUT_LW	INTS_WRN,<R0,#0>	; Warning message
				01EF	320	10\$:			
				01EF	321		\$INTOUT_X	INTS_SUME	; Source update merge error
				01F5	322		RSB		

```

01F6 324 .SBTTL OPEN NEXT INPUT SOURCE FILE
01F6 325
01F6 326 :++
01F6 327 :
01F6 328 : THIS ROUTINE IS CALLED TO OPEN THE NEXT INPUT FILE.
01F6 329 : IF THE FILE IS SUCCESSFULLY OPENED AND CONNECTED,
01F6 330 : R0 IS SET TO 1. IF ANOTHER INPUT FILE CANNOT BE OPENED,
01F6 331 : R0 IS 0.
01F6 332 :--
01F6 333
01F6 334 MAC$NXTINPFIL:
01F6 335 PUSHR #^M<R1,R2,R3,R4,R5> ;SAVE REGISTERS USED BY MOVCS
00000000'GF 01 9F 01F8 336 PUSHAB W^MAC$GT SCB ; Supply SUM control block address
50 0000'CF 01 FB 01FC 337 CALLS #1,G^SUM$CLOSE ; Close any update files
50 0000'CF 01 D0 0203 338 $DISCONNECT RAB=W^MAC$INPUT RAB ;DISCONNECT THE RECORD ACCESS
27 20 0000'CF 01 D0 020E 339 MOVL W^MAC$GL CURINFDB,R0 ;POINT TO CURRENT FDB
0000'CF 01 D0 0213 340 $CLOSE FAB=8(R0) ;CLOSE THE INPUT FILE
0000'CF 01 D0 021D 341 MOVL W^MAC$GL LINENUM,- ;SET LINE BASE TO HIGHEST LINE NUMBER
27 20 6E 00 2C 0221 342 W^MAC$GL LINBAS ;
0000'CF 01 D0 0224 343 MOVCS #0,(SP),#^A/ /,#39,- ;BLANK FILL THE RESULTANT FILENAME
50 0000'CF 01 D0 0229 344 W^MAC$AB SBT_FILE ;
0004'CF 01 50 D0 022C 345 MOVL W^MAC$GL CURINFDB,R0 ;GET ADDRESS OF FDB WE JUST CLOSED
50 0000'CF 01 50 D1 0231 346 CMLR R0,W^MAC$GL_INPQUE+4 ;WAS IT THE LAST FDB
50 09 13 0236 347 BEQL 20$ ;IF EQL YES
50 60 D0 0238 348 MOVL (R0),R0 ;NO--LINK TO NEXT FDB
FDC2' 30 023B 349 BSBW MAC$OPEN INPUT ;OPEN NEXT INPUT FILE
3E BA 023E 350 POPR #^M<R1,R2,R3,R4,R5> ;RESTORE REGISTERS
05 0240 351 RSB ;RETURN WITH LBS IN R0
0241 352 :
0241 353 : NO MORE INPUT FILES
0241 354 :
50 D4 0241 355 20$: CLRL R0 ;INDICATE NO MORE FILES
3E BA 0243 356 POPR #^M<R1,R2,R3,R4,R5>
05 0245 357 RSB

```

```

0246 359          .SBTTL  STAT4          SWITCH INPUT TO MACRO TEXT
0246 360
0246 361 :++
0246 362 : FUNCTIONAL DESCRIPTION:
0246 363 :
0246 364 : THIS SEMANTIC ROUTINE SWITCHES THE POINTER TO THE CURRENT
0246 365 : INPU BLOCK TO POINT TO THE MACRO BEING EXPANDED. FIRST THE
0246 366 : REAL MACRO ARGUMENTS ARE SCANNED AND AN INPUT BLOCK IS CREATED.
0246 367 : THEN MAC$GETCHR WILL READ CHARACTERS FROM THE MACRO TEXT
0246 368 : EXPANDER MAC$GET_MAC_LIN.
0246 369 :
0246 370 :--
0246 371
00000000 372          .PSECT  MAC$RO_CODE_P1,NOWRT,GBL,LONG
OE 0005'CF  E8 0000 373 STAT4::          ;STATEMENT = MACTXT
56 0000'CF47 D0 0013 374          BLBS      W^LST$G_MACROCALL+SYMSL VAL,10$ ;BRANCH IF LISTING MACRO CALLS
55 00'8F 9A 0019 375          $INTOUT LW INT$_SETLONG,<#0,#MAC$G_LLIST IT> ;NO--SEND FLAG TO PASS 2
      FFE0' 30 001D 376 10$:      MOVL     W^MAC$AC_VALSTACK[R7],R6 ;GET PTR TO MACRO MNB
      05 A6 D5 0020 377          MOVZBL  #CRFSK REF,R5 ;THIS IS A REFERENCE
      1D 13 0023 378          BSBW     MAC$CREF MACRO ;CROSS REFERENCE IF CREFFING
0000'CF 56 D0 0025 379          TSTL     MNBSL_TOTP(R6) ;IS THERE ANY TEXT
0000'CF 18 A6 D0 002A 380          BEQL     20$ ;IF EQL NO--TAKE THE QUICK OUT
      FFC0' 30 0030 381          MOVL     R6,W^MAC$G_L MACPTR ;SAVE POINTER TO MACRO MNB
      0033 382          MOVL     MNBSL_ARGP(R6),W^MAC$G_L_KEYMAC ;SET PTR TO KEYWORD ARGS (IF ANY)
0000'CF 0000'CF D4 0033 383          BSBW     MAC$GET_ARGS ;SCAN REAL MACRO ARGS
      0037 384          ;AND SET UP INPUT BLOCK
00 6B 10 E3 003E 385          CLRL     W^MAC$G_L_KEYMAC ;CLEAR POINTER TO KEYWORD ARGS
      5A 0D 9A 0042 386          MOVL     W^MAC$G_L_BLKPTR,W^MAC$G_L_INPUTP ;POINT TO NEW INPUT BLOCK
      05 0045 387          BBCS     #FLG$V_MACTXT,(R11),20$ ;FLAG READING MACRO TEXT
      20$:      MOVZBL  #CR,R10 ;FORCE READING OF NEW LINE
      30$:      RSB

```

```

0046 391          .SBTTL MEXIT          MACRO EXIT ROUTINE
0046 392
0046 393 :++
0046 394 : FUNCTIONAL DESCRIPTION:
0046 395 :
0046 396 : THIS ROUTINE POPS THE INPUT CONTEXT ONE LEVEL TO EFFECT AN
0046 397 : EXIT FROM A MACRO OR REPEAT-TYPE MACRO
0046 398 :
0046 399 :--
0046 400
0046 401 MEXIT::          ;DIRECTIVE = KMEXIT
0046 402
0046 403          .SBTTL MAC$POP_INPUT    POP INPUT CONTEXT UP A LEVEL
0046 404
0046 405 :++
0046 406 : FUNCTIONAL DESCRIPTION:
0046 407 :
0046 408 : THIS ROUTINE POPS THE INPUT CONTEXT A LEVEL. ALL PAGES
0046 409 : ALLOCATED TO THE CURRENT INPUT BLOCK ARE DEALLOCATED.
0046 410 :
0046 411 :--
0046 412
0046 413 MAC$POP_INPUT::
0046 414          PUSHL R6          ;SAVE R6
18 68 10 50 14 A6 56 DD 0046 414          PUSHL R6          ;SAVE R6
0000'CF DD 0048 415          MOVL W^MAC$GL_INPUTP,R6      ;GET POINTER TO CURRENT INPUT BLOCK
18 68 10 50 14 A6 56 DD 0048 415          MOVL W^MAC$GL_INPUTP,R6      ;GET POINTER TO CURRENT INPUT BLOCK
0000'CF DD 004D 416          BBC #FLG$V_MACTXT,(R11),10$ ;BRANCH IF NOT READING MACRO TEXT
0000'CF DD 0051 417          MOVL INP$RPTCNT(R6),R0      ;YES--GET REPEAT COUNT (OR MACRO FLAG)
0000'CF DD 0055 418          BGEQ 10$          ;IF GEQ NO NEED TO GO ANY FARTHER
0000'CF DD 0057 419          INCL R0          ;SEE IF IT WAS REPEAT THAT WENT TO 0
0000'CF DD 0059 420          BEQL 10$          ;IF EQL YES
0000'CF DD 005B 421          ;NO--IT WAS MACRO REDEFINING ITSELF
56 14 A6 80000000 8F CB 005B 422          BICL3 #^X80000000,INP$RPTCNT(R6),R6 ;YES--GET ADDRESS OF MNB
0000'CF DD 0064 423          BSBW MAC$DEL_MAC_DEF ;AND DELETE THE MNB AND ASSOCIATES
0000'CF DD 0067 424          MOVL W^MAC$GL_INPUTP,R6      ;RE-GET THE INPUT BLOCK ADDRESS
0000'CF DD 006C 425 10$: MOVL INP$LINK(R6),R0      ;POP INPUT CONTEXT
0000'CF DD 006F 426          MOVL R0,W^MAC$GL_INPUTP ;
0000'CF DD 0074 427          CMPL R0,#MAC$GL_P$MINBL ;READING FROM SOURCE FILE?
0000'CF DD 007B 428          BNEQ 20$          ;IF NEQ NO
0000'CF DD 007D 429          BBCC #FLG$V_MACTXT,(R11),20$ ;YES--CLEAR MACTXT FLAG
0000'CF DD 0081 430 20$: MOVL INP$_IFVAL(R6),W^MAC$GL_IF_VALUE ;POP IF VALUE
0000'CF DD 0087 431          MOVL INP$_IFLVL(R6),W^MAC$GL_IF_LEVEL ;POP IF LEVEL
0000'CF DD 008D 432          MOVL R6,R0          ;COPY INPUT BLOCK ADDRESS
56 18 A6 56 DD 0090 433          MOVL INP$PAGP(R6),R6      ;LINK TO ANY OTHER PAGES
FF69' DD 0094 434          BSBW MAC$DEA_1_PAGE ; Deallocate block (always 1 page)
06 11 0097 435          BRB 40$          ;GO DEALLOCATE INPUT BLOCK AND REST
0099 436          ;OF INPUT CONTEXT PAGES
0099 437 30$:
56 66 DD 0099 438          MOVL MXBSL_LINK(R6),R6 ; Link to possible next page
FF61' DD 009C 439          BSBW MAC$DEAL_BLOCK ; Deallocate memory block
009F 440 40$:
50 56 DD 009F 441          MOVL R6,R0          ;POINT R0 FOR NEXT DEALLOCATION
F5 12 00A2 442          BNEQ 30$          ;IF NEQ GO DO IT
5A 0D 9A 00A4 443          MOVZBL #CR,R10      ;FORCE NEW LINE
56 8ED0 00A7 444          POPL R6          ;RESTORE R6
05 00AA 445          RSB

```

```

00AB 447      .SBTTL STATEMENTS      END-OF-LINE CLEANUP
00AB 448
0CAB 449      :++
00AB 450      :
00AB 451      :--
00AB 452
00AB 453      :MTXT1::
00AB 454      :MTXT2::
00AB 455      MTXT3::
0CAB 456      MTXT4::
00AB 457      MTEXT:
00AB 458
00AB 459      $INTOUT_X INT$ CHKL
00B1 460      BISL2  #FLGSM_BOL!FLGSM_EVALEXP!FLGSM_COMPEXP,- ;RESET BEGINNING OF LINE
00B7 461      (R11)
00B8 462      CLRL  W^MAC$GL_ABSFLAG ;RESET ABSOLUTE FLAG
00BC 463      CLRL  W^MAC$GL_PRMSEG ;NO EXPRESSION PSECT YET
00C0 464      CLRL  W^MAC$GL_MOPPTR ;CLEAR POINTER TO OPERAND MODE BYTES
6B 01802000 8F CA 00C4 465      BICL2  #FLGSM_OPRND!FLGSM_IFSTAT!FLGSM_NOREF,(R11) ;NOT IN OPERAND FIELD
00CB 466      ;AND NOT IN AN IF
00CB 467      ;AND ALLOW PRMSYM TO REF SYMBOLS
00CB 468      MOVB  #RDX$V_DECIMAL,W^MAC$GB_RDXNDX ;RESET TO DECIMAL RADIX
05 00D0 469      RSB
00D1 470
00D1 471      .END

```

SS.TMP1 = 00000001
SS.TMP2 = 000000A0
\$COUNT = 0000003B
ARG\$K_SIZE = 000003E8
AUD\$K_SIZE = 00000010
BIT... = 00000005
BLNK = 00000020
CHR\$M_COMMA CR = 00000020
CHR\$M_ILL_CR = 00000040
CHR\$M_NUM_BER = 00000010
CHR\$M_SPA_MSK = 00000001
CHR\$M_SYM_CH1 = 00000008
CHR\$M_SYM_CHR = 00000004
CHR\$M_SYM_DLM = 00000002
CHR\$V_COMMA CR = 00000005
CHR\$V_CVTLWC = 00000061
CHR\$V_ILL_CR = 00000006
CHR\$V_NOCVT = 0000007F
CHR\$V_NUM_BER = 00000004
CHR\$V_SPA_MSK = 00000000
CHR\$V_SYM_CH1 = 00000003
CHR\$V_SYM_CHR = 00000002
CHR\$V_SYM_DLM = 00000001
CNT = 00000002
CR = 0000000D
CRFSK_REF = ***** X 04
ERR = 00000000
FF = 0000000C
FLG\$M_ALLCHR = 00000001
FLG\$M_BOL = 00000002
FLG\$M_CHKLPND = 00100000
FLG\$M_COMPEXPR = 00000004
FLG\$M_CONT = 00000008
FLG\$M_CRF = 40000000
FLG\$M_CRSEEN = 00000001
FLG\$M_DATRPT = 00000010
FLG\$M_DBGOUT = 00004000
FLG\$M_DLIMSTR = 00008000
FLG\$M_ENDMCH = 00000020
FLG\$M_EVALEXPR = 00000040
FLG\$M_EXPOPT = 00000080
FLG\$M_EXTERR = 00010000
FLG\$M_EXTWRN = 00020000
FLG\$M_FIRSTLN = 00000200
FLG\$M_IFSTAT = 00800000
FLG\$M_IIF = 00400000
FLG\$M_INSERT = 00000100
FLG\$M_IRPC = 20000000
FLG\$M_LEXOP = 00000002
FLG\$M_LSTXST = 00000200
FLG\$M_MAC2COL = 00000800
FLG\$M_MACL = 0C000800
FLG\$M_MACLYB = 08000000
FLG\$M_MACTXT = 00010000
FLG\$M_MEBLST = 00001000
FLG\$M_MOREARG = 00002000
FLG\$M_MOREINP = 00000008

FLG\$M_NEWPND = 00000400
FLG\$M_NOREF = 01000000
FLG\$M_NTTYPEPC = 00000020
FLG\$M_NULCHR = 00040000
FLG\$M_OBJXST = 00200000
FLG\$M_OPNDCHK = 00000100
FLG\$M_OPRND = 00002000
FLG\$M_OPTVFLIDX = 00001000
FLG\$M_ORDLST = 00020000
FLG\$M_P2 = 00004000
FLG\$M_RPTIRP = 10000000
FLG\$M_SEQFIL = 02000000
FLG\$M_SKAN = 00008000
FLG\$M_SPECOP = 00000004
FLG\$M_SPLALL = 04000000
FLG\$M_STOIMF = 00040000
FLG\$M_SYM2COL = 00000400
FLG\$M_TOCF LG = 00080000
FLG\$M_UPAF LG = 00000010
FLG\$M_UPDFIL = 00000080
FLG\$M_UPMARG = 00000040
FLG\$M_XCRF = 80000000
FLG\$V_ALLCHR = 00000000
FLG\$V_BOL = 00000001
FLG\$V_CHKLPND = 00000014
FLG\$V_COMPEXPR = 00000002
FLG\$V_CONT = 00000003
FLG\$V_CRF = 0000001E
FLG\$V_CRSEEN = 00000020
FLG\$V_DATRPT = 00000004
FLG\$V_DBGOUT = 0000002E
FLG\$V_DLIMSTR = 0000002F
FLG\$V_ENDMCH = 00000005
FLG\$V_EVALEXPR = 00000006
FLG\$V_EXPOPT = 00000007
FLG\$V_EXTERR = 00000030
FLG\$V_EXTWRN = 00000031
FLG\$V_FIRSTLN = 00000029
FLG\$V_IFSTAT = 00000017
FLG\$V_IIF = 00000016
FLG\$V_INSERT = 00000008
FLG\$V_IRPC = 0000001D
FLG\$V_LEXOP = 00000021
FLG\$V_LSTXST = 00000009
FLG\$V_MAC2COL = 0000002B
FLG\$V_MACL = 0000000B
FLG\$V_MACLTB = 0000001B
FLG\$V_MACTXT = 00000010
FLG\$V_MEBLST = 0000000C
FLG\$V_MOREARG = 0000002D
FLG\$V_MOREINP = 00000023
FLG\$V_NEWPND = 0000000A
FLG\$V_NOREF = 00000018
FLG\$V_NTTYPEPC = 00000025
FLG\$V_NULCHR = 00000032
FLG\$V_OBJXST = 00000015
FLG\$V_OPNDCHK = 00000028


```

FLGSV_OPRND      = 0000000D
FLGSV_OPTVFLIDX = 0000002C
FLGSV_ORDLST    = 00000011
FLGSV_P2        = 0000000E
FLGSV_RPTIRP    = 0000001C
FLGSV_SEQFIL    = 00000019
FLGSV_SKAN      = 0000000F
FLGSV_SPECOP    = 00000022
FLGSV_SPLALL    = 0000001A
FLGSV_STOIMF    = 00000012
FLGSV_SYM2COL   = 0000002A
FLGSV_TOCLG    = 00000013
FLGSV_UPAFIL    = 00000024
FLGSV_UPDFIL    = 00000027
FLGSV_UPMARG    = 00000026
FLGSV_XCRF      = 0000001F
HASHSZ          = 0000007F
HYPHEN          = 0000002D
INPSB_ARGCT     = 0000001C
INPSK_BLKSIK   = 00000021
INPSK_BUFSIZ    = 000003E8
INPSK_IRPSIZ    = 0000003C
INPSL_ARGS      = 0000001D
INPSL_GETL      = 00000008
INPSL_IFLVL     = 0000000C
INPSL_IFVAL     = 00000010
INPSL_LINK      = 00000000
INPSL_NXTL      = 00000004
INPSL_PAGP      = 00000018
INPSL_RPTCNT    = 00000014
INTSK_BUFSIZ    = 000013F4
INTSK_BUFWRN    = 00001390
INTS_ADD        = 00000001
INTS_AND        = 00000002
INTS_ASH        = 00000003
INTS_ASN        = 0000000C
INTS_AUGPC      = 0000000D
INTS_BDST       = 0000000E
INTS_CHKL       = 0000000F
INTS_DIV        = 00000004
INTS_END        = 00000010
INTS_EPT        = 00000011
INTS_ERR        = 00000012
INTS_ETX        = 00000013
INTS_FNEWL      = 00000014
INTS_ILG        = 00000000
INTS_INFO       = 0000003A
INTS_LGLAB      = 00000015
INTS_MACL       = 00000016
INTS_MUL        = 00000005
INTS_NEG        = 00000006
INTS_NEWL       = 00000017
INTS_NEWP       = 00000018
INTS_NOT        = 00000007
INTS_OP         = 00000019
INTS_OR         = 00000008
INTS_PRIL       = 0000001A

```

```

INTS_PRT        = 0000001B
INTS_PSECT      = 0000001C
INTS_REDEF      = 0000001D
INTS_REF        = 0000001E
INTS_REST       = 0000001F
INTS_SAME       = 00000009
INTS_SAVE       = 00000020
INTS_SBTTL      = 00000021
INTS_SETFLAG    = 00000022
INTS_SETLONG    = 00000023
INTS_SPIC       = 00000024
INTS_SPID       = 00000025
INTS_STIB       = 00000026
INTS_STIL       = 00000028
INTS_STIW       = 00000027
INTS_STKEPT     = 00000029
INTS_STKG       = 0000002A
INTS_STKL       = 0000002B
INTS_STKPC      = 0000002C
INTS_STKS       = 0000002D
INTS_STCB       = 00000034
INTS_STOL       = 0000002E
INTS_STOW       = 00000035
INTS_STRB       = 0000002F
INTS_STRL       = 00000031
INTS_STRSB      = 00000032
INTS_STRSW      = 00000033
INTS_STRW       = 00000030
INTS_STSB       = 00000036
INTS_STSW       = 00000037
INTS_SUB        = 0000000A
INTS_SUM        = 00000039
INTS_WRN        = 00000038
INTS_XOR        = 0000000B
LST$G_MACROCALL ***** X 04
LST$K_BUFSIZ    = 00000086
LST$K_L_P_PAGE = 0000003C
LST$K_TITLE_SIZ = 00000028
MAB$B_ARGNO     = 00000005
MAB$B_NAME      = 00000004
MAB$K_BLKSIK   = 0000000C
MAB$K_DVPTTR   = 00000008
MAB$K_LINK      = 00000000
MAB$K_DVLEN     = 00000006
MAC$AB_LINEBF   ***** X 03
MAC$AB_SBT_FILE ***** X 03
MAC$AL_VALSTACK ***** X 04
MAC$CREF_MAC O ***** X 04
MAC$DEAL_BLOCK ***** X 04
MAC$DEA_T_PAGE ***** X 04
MAC$DEL_MAC_DEF ***** X 04
MAC$GB_RDXNDX   ***** X 04
MAC$GETCHR      = 00000000 RG 03
MAC$GETLIN      = 000000BB RG 03
MAC$GET_ARGS    ***** X 04
MAC$GL_ABSFLAG ***** X 04
MAC$GL_BLKPTR   ***** X 04

```

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
. ABS :	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
. BLANK :	00000000 (0.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
\$AB\$\$	0000003C (60.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
MAC\$RO_CODE_P15	00000246 (582.)	03 (3.)	NOPIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC LONG
MAC\$RO_CODE_P1	000000D1 (209.)	04 (4.)	NOPIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC LONG

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.02	00:00:01.67
Command processing	107	00:00:00.48	00:00:04.12
Pass 1	263	00:00:04.89	00:00:24.79
Symbol table sort	0	00:00:00.65	00:00:02.10
Pass 2	102	00:00:01.14	00:00:04.59
Symbol table output	50	00:00:00.23	00:00:00.43
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	555	00:00:07.43	00:00:37.72

The working set limit was 1500 pages.
42445 bytes (83 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 652 non-local and 38 local symbols.
471 source lines were read in Pass 1, producing 22 object records in Pass 2.
25 pages of virtual memory were used to define 23 macros.

! Macro library statistics !

Macro library name	Macros defined
-\$255\$DUA28:[SHRLIB]SUM.MLB;1	3
-\$255\$DUA28:[MACRO.OBJ]MACRO.MLB;1	12
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	9
TOTALS (all libraries)	24

827 GETS were required to define 24 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:INPUT/OBJ=OBJ\$:INPUT MSRC\$:INPUT/UPDATE=(ENH\$:INPUT)+LIB\$:MACRO/LIB+SHRLIB\$:SUM/LIB

The image displays a grid of 100 small terminal window screenshots, arranged in a 10x10 grid. Each window shows a different VAX/VMS command and its output. The windows are arranged in a grid, with some windows clearly legible and labeled with their command names. The labels are as follows:

- DATA LIS (top-left)
- DEFINE LIS (middle-left)
- FLOAT LIS (middle-right)
- ERRMSG LIS (lower-middle)
- GETARG LIS (lower-right)
- DATA LIS (bottom-left)
- INPUT LIS (bottom-right)
- ERROR LIS (bottom-center)
- FINISH LIS (bottom-center)
- GETCMD LIS (bottom-center)

The screenshots show various system outputs, including command prompts, file listings, and system messages. The text is small and dense, typical of a terminal window output.

Terminal 1	Terminal 2	Terminal 3	Terminal 4	Terminal 5	Terminal 6	Terminal 7	Terminal 8	Terminal 9	Terminal 10	Terminal 11	Terminal 12
Terminal 13	Terminal 14	Terminal 15	Terminal 16	Terminal 17	Terminal 18	Terminal 19	Terminal 20	Terminal 21	Terminal 22	Terminal 23	Terminal 24
Terminal 25	Terminal 26	Terminal 27	Terminal 28	Terminal 29	Terminal 30	Terminal 31	Terminal 32	Terminal 33	Terminal 34	Terminal 35	Terminal 36
Terminal 37	Terminal 38	Terminal 39	Terminal 40	Terminal 41	Terminal 42	Terminal 43	Terminal 44	Terminal 45	Terminal 46	Terminal 47	Terminal 48
Terminal 49	Terminal 50	Terminal 51	Terminal 52	Terminal 53	Terminal 54	Terminal 55	Terminal 56	Terminal 57	Terminal 58	Terminal 59	Terminal 60
Terminal 61	Terminal 62	Terminal 63	Terminal 64	Terminal 65	Terminal 66	Terminal 67	Terminal 68	Terminal 69	Terminal 70	Terminal 71	Terminal 72
Terminal 73	Terminal 74	Terminal 75	Terminal 76	Terminal 77	Terminal 78	Terminal 79	Terminal 80	Terminal 81	Terminal 82	Terminal 83	Terminal 84
Terminal 85	Terminal 86	Terminal 87	Terminal 88	Terminal 89	Terminal 90	Terminal 91	Terminal 92	Terminal 93	Terminal 94	Terminal 95	Terminal 96
Terminal 97	Terminal 98	Terminal 99	Terminal 100	Terminal 101	Terminal 102	Terminal 103	Terminal 104	Terminal 105	Terminal 106	Terminal 107	Terminal 108
Terminal 109	Terminal 110	Terminal 111	Terminal 112	Terminal 113	Terminal 114	Terminal 115	Terminal 116	Terminal 117	Terminal 118	Terminal 119	Terminal 120
Terminal 121	Terminal 122	Terminal 123	Terminal 124	Terminal 125	Terminal 126	Terminal 127	Terminal 128	Terminal 129	Terminal 130	Terminal 131	Terminal 132
Terminal 133	Terminal 134	Terminal 135	Terminal 136	Terminal 137	Terminal 138	Terminal 139	Terminal 140	Terminal 141	Terminal 142	Terminal 143	Terminal 144